1976

Finite element solution of the nonlinear coupled neutronic-energy equations for a fast reactor fuel cell.

Kasdorf, Roy Edward

Monterey, California. Naval Postgraduate School

FINITE ELEMENT SOLUTION OF THE NONLINEAR COUPLED NEUTRONIC-ENERGY EQUATIONS FOR A FAST REACTOR FUEL CELL

Roy Edward Kasdorf

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

FINITE ELEMENT SOLUTION OF THE NONLINEAR
COUPLED NEUTRONIC-ENERGY EQUATIONS
FOR A FAST REACTOR FUEL CELL

by

Roy Edward Kasdorf

December 1976

Thesis Advisor: D. Nguyen
Thesis Advisor: D. Salinas

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Finite Element Solution of the Non-linear Coupled Neutronic-Energy Equations for a Fast Reactor Fuel Cell		5. TYPE OF REPORT & PERIOD COVERED Master's & Mechanical Engineer's Thesis; December 1976
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Roy Edward Kasdorf		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE December 1976
		13. NUMBER OF PAGES 155
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Naval Postgraduate School Monterey, California 93940		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A transient overpower (TOP) accident in a Liquid Metal Fast Breeder Reactor (LMFBR) is considered. The analysis is formulated to model the dynamic response of the reactor fuel subassembly during the initial period of the postulated overpower transient. An equivalent cylindrical cell is used to model the fuel subassembly. The governing neutronic and heat transport equations for each region (fuel, clad, and

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
(Page 1) S/N 0102-014-6601 |

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

coolant) of the equivalent cylindrical cell are developed. Nuclear Doppler broadening feedback is included in the dynamic model making the coupled equations non-linear. The resulting non-linear partial differential field equations are transformed into a system of ordinary differential equations by the finite element method. An isoparametric, quadratic, rectangular element is used for the discretization of the spatial domain. When using the finite element method, large system matrices may result. To facilitate solution of these large systems, an optimum compacting scheme is utilized. The implicit Gear's method is used for the solution of the system of ordinary differential equations. The results for a sample problem are presented.

Finite Element Solution of the Nonlinear
Coupled Neutronic-Energy Equations
for a Fast Reactor Fuel Cell

by

Roy Edward Kasdorf
Lieutenant, United States Nayv
B.S.M.E., New Mexico State University, 1970

Submitted in partial fulfillment of the
requirements for the degrees of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

and

MECHANICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
December 1976

ABSTRACT

A transient overpower (TOP) accident in a Liquid Metal Fast Breeder Reactor (LMFBR) is considered. The analysis is formulated to model the dynamic response of the reactor fuel subassembly during the initial period of the postulated overpower transient. An equivalent cylindrical cell is used to model the fuel subassembly. The governing neutronic and heat transport equations for each region (fuel, clad, and coolant) of the equivalent cylindrical cell are developed. Nuclear Doppler broadening feedback is included in the dynamic model making the coupled equations non-linear. The resulting non-linear partial differential field equations are transformed into a system of ordinary differential equations by the finite element method. An isoparametric, quadratic, rectangular element is used for the discretization of the spatial domain. When using the finite element method, large system matrices may result. To facilitate solution of these large systems, an optimum compacting scheme is utilized. The implicit Gear's method is used for the solution of the system of ordinary differential equations. The results for a sample problem are presented.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND NOTATION

A. NOTATION

$< \quad >$	Row vector
$\{ \quad \}$	Column vector
$[\quad]$	Square matrix or indicates a reference
$[\quad]^{-1}$	Inverse of a square matrix
∇	Del operator
$\frac{\partial}{\partial x}$	Partial derivative with respect to x
Δx	Change in x
$\int_V$	Volume integral
$\int_x$	Integration with respect to x
det [x]	Determinant of x

B. SYMBOLS

b	Nuclear Doppler constant	
C	Concentration of delayed neutron precursors	
C_p	Specific heat	[cal/gm °C]
$D(\underline{r})$	Neutron diffusion coefficient	[cm]
e	Nuclear energy released per fission	[cal/fission]
h	Heat transfer coefficient	[cal/cm^{2}sec °C]
J	Jacobian matrix	
$J(\underline{r},t)$	Neutron current	$[\frac{\text{neutrons}}{\text{cm}^2\ \text{sec}}]$
k_∞	Infinite multiplication factor	
K_D	Doppler constant	
$k(\underline{r})$	Thermal conductivity	[cal/cm sec °C]
LMFBR	Liquid Metal Fast Breeder Reactor	

LOCA	Loss of Coolant Accident	
N	Shape function	
n	Number of delay neutron groups	
$n(\underline{r},t)$	Neutron density	$[\frac{\text{neutrons}}{\text{cm}^3}]$
N,x	Derivative of N with respect to x	
$\dot{q}(\underline{r},t)$	Nuclear generation	$[\text{cal/cm}^3\text{sec}]$
R	Residual	
$\underline{r}$	Spatial coordinate	[cm]
r,z	Global coordinates	
$S(\underline{r},t)$	Neutron production	$[\frac{\text{neutrons}}{\text{cm}^3\ \text{sec}}]$
T	Temperature	[°C]
t	Time	[sec]
TOP	Transient Overpower	
v	Neutron velocity	[cm/sec]
V_{co}	Velocity of coolant flow	[cm/sec]
W	Weighting function	
β	Fraction of fission neutrons which appear as delayed neutrons	
$\phi(\underline{r},t)$	Neutron flux	$[\frac{\text{neutrons}}{\text{cm}^2\ \text{sec}}]$
λ	Decay constant of the delayed neutron precursors	[1/sec]
η,ξ	Local coordinates	
ν	Average number of neutrons released per fission	
ρ	Reactivity	
$\rho(\underline{r})$	Density	$[\text{gm/cm}^2]$
Σ	Neutron cross section	$[\text{cm}^{-1}]$

C. SUBSCRIPTS

a	Absorption
c	Clad
co	Coolant
CR	Critical
D	Delayed, Doppler
f	Fission
F	Fuel
gap	Fuel-clad interface
i,j	Group, equation
P	Prompt
surf	Clad-coolant interface

D. SUPERSCRIPTS

e	Element
e*	Adjacent element
o	At time zero
.	Derivative with respect to time

ACKNOWLEDGEMENTS

The author wishes to express his appreciation to Dr. Dong Nguyen and to Dr. David Salinas, Professors of Mechanical Engineering, for their advice and guidance throughout the course of this work.

The author is obligated to Dr. Richard Franke, Professor of Mathematics, for his assistance in implementing the integration procedure and for his assistance as second reader.

The author wishes to thank Dr. Gilles Cantin, Professor of Mechanical Engineering, for his invaluable discussions concerning the finite element method.

Finally, the author wishes to thank his wife, Gail, for her encouragement and understanding throughout the course of this study.

I. INTRODUCTION

As the world's fossil fuel resources are depleted, more emphasis is being placed on the breeder reactor as a potential means of solving the coming energy crisis. While the development of new energy sources is being pushed, equal effort is being given to the maintenance of an environmentally clean world. To this end, the safety of breeder reactors is receiving a considerable amount of attention before assuming that the breeder reactor is the answer to the energy problem.

The Liquid Metal Fast Breeder Reactor (LMFBR) appears to be one of the most promising breeder reactors. Most engineers will concede there is little probability of a nuclear explosion occurring in the operation of a nuclear reactor. Of major concern to engineers is the loss of coolant accident (LOCA) and the transient overpower accident (TOP). The present analysis is concerned with a TOP accident in a LMFBR. The analysis is formulated to model the dynamic response of the reactor fuel subassembly during the initial period of the postulated overpower transient. The primary consideration is given to the early response of this fuel subassembly to various conditions of disturbances. The phenomenon which occurs after core disassembly (i.e., clad melting) is not the concern of this analysis. Only the time prior to clad melting is being considered.

No consideration is given here as to how the overpower transient occurs or to why the safety features of the reactor did not operate properly. It is postulated that the accident has occurred. In this analysis, the TOP accident is created by either a step increase in reactivity, a ramp increase in reactivity, or a combination of both.

An inherent safety feature of most reactors, nuclear Doppler broadening feedback, is included in the dynamic model of the fuel subassembly. The Doppler feedback acts to reduce the effect of the excursion. Consideration of this feedback creates a non-linear system model which is described by a non-linear, initial-boundary-value problem.

The conventional method of solution uses the standard point kinetics formulation. Recent studies have pointed out a non-negligible error in this model [1], particularly with asymmetric disturbances [2], or space-dependent feedback [3]. In Ref. [4], a somewhat novel approach of using the finite element method (FEM) for the space-time dependent solution of the reactor dynamics problem was demonstrated. The FEM is effective in handling these asymmetric disturbances and space-dependent feedbacks. Therefore, the finite element method was used so that the spatial effects on the postulated problem may be studied further.

The purpose of this work was to demonstrate further the applicability of the FEM to the non-linear reactor dynamics problem as well as to investigate the dynamic response of the reactor fuel subassembly. The analysis required a novel

approach to handle the gap conductances present at the interfaces of the equivalent cell model of the fuel subassembly; this will be clarified in the analysis.

II. DESCRIPTION OF PROBLEM

A. PHYSICAL SYSTEM

The typical Liquid Metal Fast Breeder Reactor (LMFBR) core consists of many hexagonal modules, each containing several hundred fuel pins. For this analysis, an equivalent cylindrical cell is used to model the fuel subassembly; see Figure 1. The use of equivalent cells as models for larger systems has been common practice in nuclear analysis (i.e., the well known Wigner-Sietz method). In using an equivalent cell, the actual shape of the reactor core is not important, and the analysis is applicable to any reactor which has the same equivalent cell.

The equivalent cell considered in this analysis, Figure 1, is fueled with enriched uranium dioxide, has a stainless steel cladding, and has liquid sodium for a coolant. The dimensions used are

$$a = 0.254 \text{ cm},$$

$$b = 0.292 \text{ cm},$$

$$c = 0.365 \text{ cm},$$

and

$$H = 33.0 \text{ cm}.$$

The gap between the fuel and cladding is very small and, in fact, may be nonexistent as in bonded fuels. The dimension of this gap has been assumed negligible. The height, H, of the fuel rod is shorter than many proposed systems (Fast Flux Testing Facility and Clinch River Breeder Reactor). However,

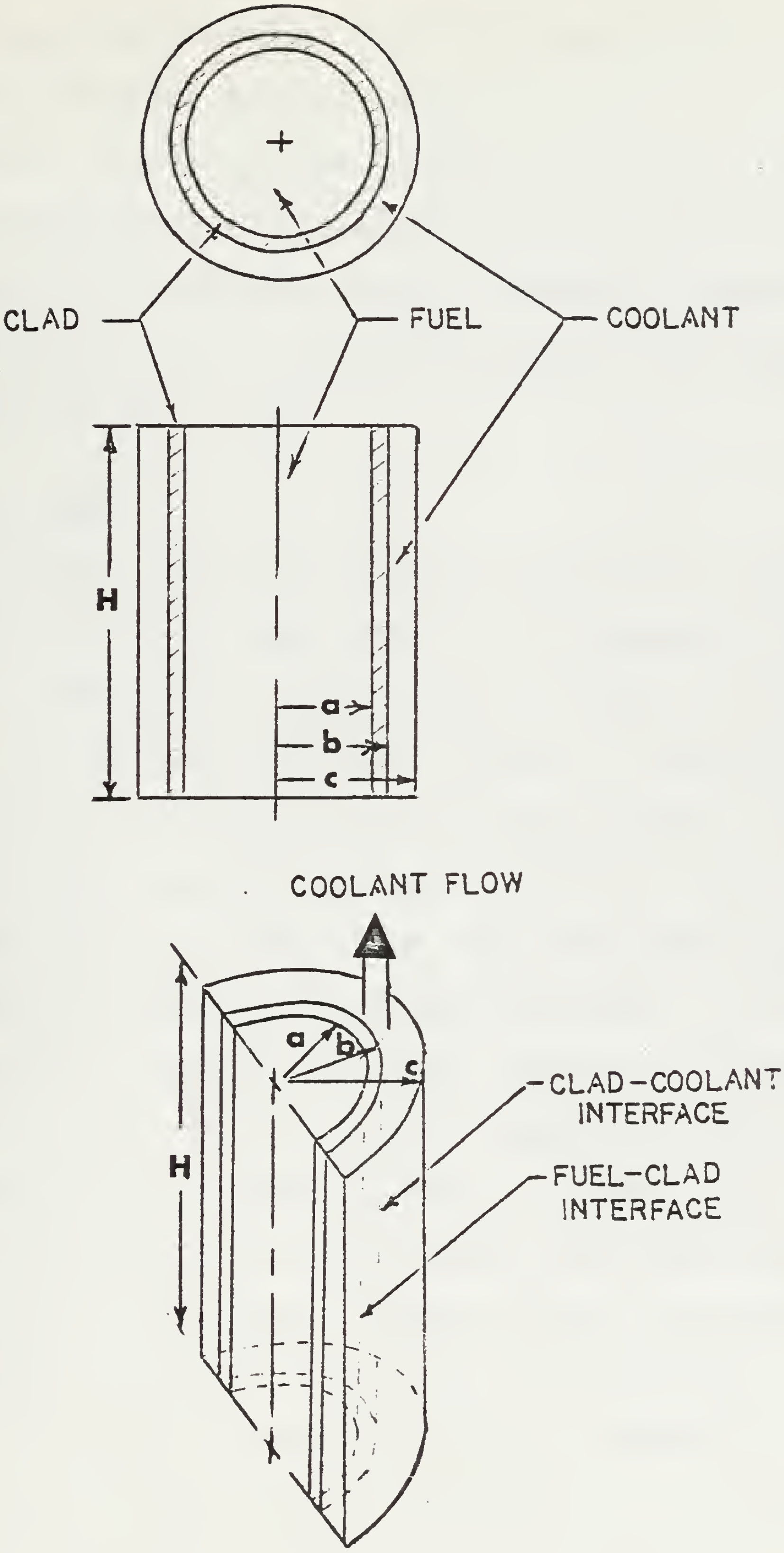


Figure 1. Equivalent Cylindrical Cell

to facilitate the numerical solution a smaller rod was used. The dynamic behavior prior to fuel pin failure for this system should be similar to the behavior of larger systems.

The treatment of this problem in three dimensions would be prohibitive in computer usage. Therefore, azimuthal symmetry is assumed, and the problem becomes a two-dimensional cylindrical (r,z) problem.

B. SYSTEM MODEL

The analysis considers the monoenergetic neutron diffusion approximation to model the transient neutron transport problem. A simple conduction-convection heat transfer model is used for the energy transport problem. The temperatures in the model are directly coupled to the neutron flux through the nuclear heat generation within the fuel. The neutron population is in turn coupled to the temperature through any of a number of reactivity feedback mechanisms. The nuclear Doppler effect is perhaps the most important of these mechanisms since it provides a negative temperature coefficient which increases the inherent stability of the reactor. Prior to core disassembly and fuel melting, the nuclear Doppler effect is the most dominant feedback and is, therefore, the only feedback mechanism considered in this analysis.

For irradiated, mixed-oxide fuels, a phenomenon of fuel restructuring has been commonly observed. This restructuring, essentially a change of phase of the fuel material, presents a unique heat transfer problem particularly during transient conditions. The problem has not been fully characterized and

is beyond the scope of this analysis. The fuel was, therefore, assumed to be a homogeneous mixture of enriched uranium dioxide.

At the fuel-cladding interface, there exists a gap which produces a thermal resistance. This thermal resistance is one of the most significant deterrents to the energy transfer to the coolant. The interface may be in physical contact or an actual gap may exist. The prediction of the thermal resistance is extremely complicated and must take into consideration many parameters: initial dimensions, type of bond, fill gas composition, fuel restructuring, fuel swelling, prior fuel life cycle, to mention a few. Reference [5] documents a computer program which attempts to predict the gap conductance, H_{gap}. This treats the thermal resistance at the interface in the same manner as a convection heat transfer coefficient when considering convection heat transfer. Since it is not the objective of this analysis to predict the gap coefficient, a representative set of values for gap coefficient, as given in Ref [6], is used in this analysis. These values are assumed to remain static during the transient. The gap conductance profile actually varies with time and will have an effect upon the transient, as noted in Ref. [7]. The prediction of this variance was not considered important for this analysis; therefore, the static assumption was made.

An average convection heat transfer coefficient was used to determine the heat transfer from the cladding to the coolant. The value used was determined from an empirical formula given in Ref. [5] and repeated in Appendix C.

In this work, consideration is given to step and ramp increases in reactivity, although any reactivity transient may easily be considered. The step and ramp increases in reactivity probably represent the most realistic physical reactivity inputs in a reactor. Once the reactivity has been inserted, the transient overpower excursion begins. Unless the Doppler feedback can override the inserted reactivity, the excursion will continue until there is physical core disassembly.

C. NUMERICAL SOLUTION

The system of equations which models the proposed problem is a non-linear, initial-boundary-value problem. The conventional method of solution of the reactor dynamics is the point kinetics formulation. It was pointed out in Refs. [1], [2], [3], and [4], that there is a non-negligible error in this model, particularly under conditions of asymmetric disturbances or space-dependent feedback. Reference [4] demonstrates the somewhat novel approach of using the finite element method (FEM) to solve the space-time dependent reactor dynamics problem. As shown in Ref. [4], the FEM is quite effective in handling localized perturbations and space-dependent feedback. In this work, only uniform disturbances were considered; however, the feedback model was space-dependent. Therefore, the finite element method is used to solve the non-linear, coupled, space-time dependent neutronic and heat transport field equations. The solution technique results in a large computer storage requirement; therefore, an optimum compact storage scheme, Ref. [8], is utilized for storage of the discretized matrices.

Once the domain has been discretized by the FEM, the solution of the resulting ordinary differential equations was to be accomplished by the implicit Gear's method, Ref. [9].

III. MODEL DEVELOPMENT

A. NEUTRONIC ANALYSIS

In this section the governing field equations for the neutron population (flux) for each of the three regions (fuel, clad, and coolant) of the domain will be formulated. The monoenergetic, diffusion theory will be used.

Consider an arbitrary volume of material within a reactor. Applying the condition of conservation to the monoenergetic neutrons leads to the neutron equation of continuity [10]

$$\frac{\partial n(r,t)}{\partial t} = S(r,t) - \Sigma_a(r)\phi(r,t) - \text{div } J(r,t) \qquad (1)$$

where

$\underline{r}$ - spatial point

t - time

$n(\underline{r},t)$ - neutron density

$S(\underline{r},t)$ - neutron production

$\Sigma_a(\underline{r})$ - neutron absorption cross section

$\phi(\underline{r},t)$ - neutron flux

$J(\underline{r},t)$ - neutron current

The left-hand side of equation (1) represents the time rate of change of the neutron density which is related to flux, ϕ , by

$$n(\underline{r},t) = \frac{1}{v}\,\phi(r,t) \qquad (2)$$

where

v - neutron velocity

On the right-hand side of equation (1), the first term is neutron production, the second term is a neutron loss through absorption, and the third term is a neutron loss through leakage from the control volume.

Using equation (2) and applying Fick's Law to the equation of continuity results in the classical neutron diffusion equation

$$\nabla \cdot (D(\underline{r}) \nabla \phi(r,t)) - \Sigma_a(\underline{r})\phi(\underline{r},t) + S(\underline{r},t) = \frac{1}{v}\frac{\partial \phi(\underline{r},t)}{\partial t} \quad (3)$$

where $D(\underline{r})$ - neutron diffusion coefficient

The vector notation used here is intended to include only two dimensions (r,z) since azimuthal symmetry has been assumed.

Equation (3) is applicable to each of the three regions of the equivalent cylindrical cell. The subscripts F (fuel), c (cladding), and co (coolant), will be used to denote these regions.

1. Fuel Region

In applying equation (3) to the fuel region the material properties of the fuel must be used. Within the fuel the neutron souce term is due to the nuclear fission process. During fission, neutrons are released as both prompt neutrons and delayed neutrons so that

$$S(\underline{r},t) = S_p(\underline{r},t) + S_D(\underline{r},t) \quad (4)$$

where $S_p(\underline{r},t)$ - prompt neutron source

$S_D(\underline{r},t)$ - delayed neutron source

The neutron sources are commonly represented as [10]

$$S_p = k_\infty \Sigma_{aF} \phi_F(1-\beta) \tag{5}$$

and

$$S_D = \sum_{i=1}^{n} C_i \lambda_i \tag{6}$$

where

k_∞ - infinite multiplication factor

β - fraction of fission neutrons which appear as delayed neutrons

n - number of delayed neutron groups

C_i - concentration of delayed neutron precursors in the ith group

λ_i - decay constant of the delayed neutrons

The space and time variables, $\underline{r}$ and t, will be dropped except where needed for clarification.

The concentration of delayed neutron precursors, C_i, is given by the following first order partial differential equation [10]

$$\frac{\partial C_i}{\partial t} = \beta_i k_\infty \Sigma_{aF} \phi_F - \lambda_i C_i \tag{7}$$

where

β_i - fraction of delayed neutrons which appear as delayed neutrons in the ith group

The solution of equation (7) is

$$C_i = \beta_i \int_0^t e^{-\lambda_i(t-t')} k_\infty(r,t')\Sigma_{aF}\phi_F dt' + C_i^\circ e^{-\lambda_i t} \qquad (8)$$

C_i° - concentration of delayed neutron precursors of the i<u>th</u> group at time zero

Reference [10] develops an expression for the initial concentration of delayed neutrons

$$C_i^\circ = \beta_i k_\infty^\circ \Sigma_{aF} \phi^\circ/\lambda_i \qquad (9)$$

where

k_∞° - initial finite multiplication factor

ϕ^o - initial neutron flux

Combining equations (3), (4), (5), (6), (8), and (9), yields the governing equation for the fuel region

$$\nabla\cdot(D_F\nabla\phi_F) + \Sigma_{aF}\phi_F[k_\infty(1-\beta) -1]$$

$$+ \sum_{i=1}^{n} \lambda_i [\beta_i \int_0^t e^{-\lambda_i(t-t')} k_\infty(\underline{r},t')\phi_F\Sigma_{aF} dt'$$

$$+ \beta_i k_\infty^\circ \Sigma_{aF}\phi_F/\lambda_i \; e^{-\lambda_i t}] = \frac{1}{v}\frac{\partial\phi_F}{\partial t} \qquad (10)$$

2. <u>Cladding Region</u>

The cladding separates the fuel and coolant and contains the fuel and the fission by-products. Equation (3) governs the neutron flux in the clad. Since the clad contains

no fissile material, the neutron source term is zero. The field equation is, then,

$$\nabla \cdot (D_c \nabla \phi_c) - \Sigma_{ac} \phi_c = \frac{1}{v} \frac{\partial \phi_c}{\partial t} \tag{11}$$

3. Coolant Region

The annular region around the cladding of the equivalent cell contains the coolant. As in the clad, there is no fissile material in the coolant and, therefore, no neutron source term. The equation governing the neutron flux in the coolant is

$$\nabla \cdot (D_{co} \nabla \phi_{co}) - \Sigma_{aco} \phi_{co} = \frac{1}{v} \frac{\partial \phi_{co}}{\partial t} \tag{12}$$

Equations (10), (11), and (12) are the one-velocity, diffusion approximation used to model the neutron transport problem.

4. Infinite Multiplication Factor

The infinite multiplication factor, k_∞, may be expressed as the infinite multiplication factor at time zero (start of transient), k_∞°, plus the postulated reactivity insertion (such as a step or a ramp), ρ, minus the change in the Doppler reactivity feedback, $\Delta\rho_D$. Other feedback mechanisms are normally not as significant as the Doppler broadening feedback prior to fuel melting and have been neglected in this analysis. Therefore,

$$k_\infty = k_\infty^\circ + \rho - \Delta\rho_D \tag{13}$$

For a fast reactor, k_∞° may be approximated as

$$k_\infty^\circ = \nu \frac{\Sigma_{fF}}{\Sigma_{aF}} \tag{14}$$

where

ν - average number of neutrons released per fission

Σ_{fF} - fission cross section of the fuel

Σ_{aF} - absorption cross section of the fuel

The nuclear Doppler effect is a very important safety feature in a nuclear reactor. Nuclei in an atom are in continual motion due to their own thermal energy. As a result of this motion, even when monoenergetic neutrons interact with the atom, there appears to be a spread in the energy of the neutron - the Doppler effect. It can be shown that the cross section of a resonance becomes less in magnitude and wider as the motion of the nuclei increases [10]. As the temperature increases, the motion increases, and the shape of a resonance cross section broadens. This broadening increases the average cross section, thus, providing a negative temperature coefficient. This effect is shown in Figure 2. It is this nuclear Doppler broadening effect which provides one of the few inherent, reliable, negative reactivity feedbacks which slows an overpower transient and possibly stops a mild overpower transient.

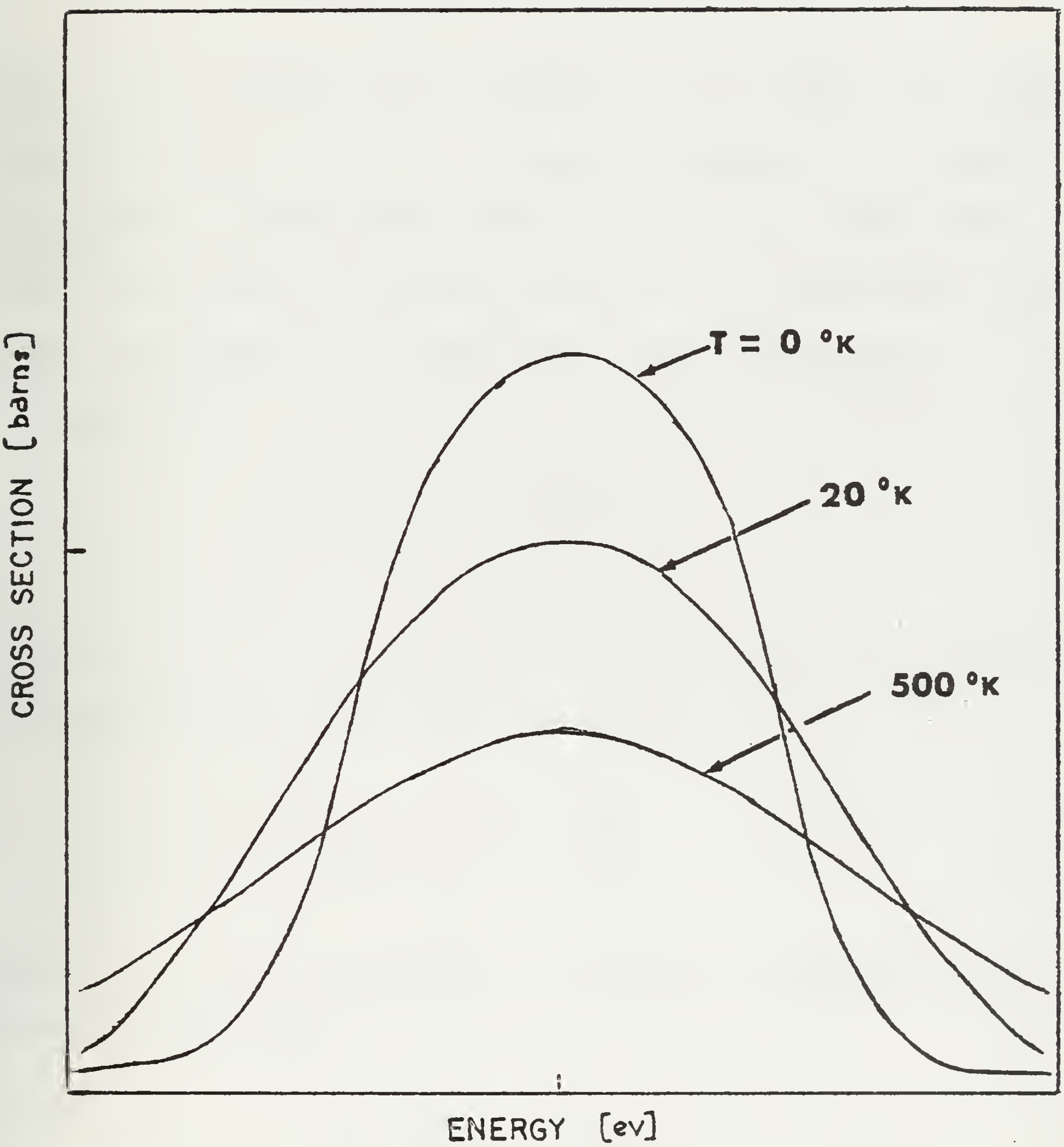


Figure 2. Doppler Broadening of a Resonance Peak

The Doppler reactivity change with respect to fuel temperature changes should be written as [6]

$$\frac{d\rho_D}{dT} = aT^{-3/2} + bT^{-1} + cT^{m-1} \qquad (15)$$

where a, b, and c are parameters determined from experimental work and m is an integer. However, as noted in Ref. [6], a substantial amount of work has shown that $T \frac{d\rho_D}{dT}$ is very nearly constant over the temperature range under consideration. Therefore, the coefficients a and c have been set equal to zero, and b is defined as

$$b = K_D = T \frac{d\rho_D}{dT} \qquad (16)$$

The constant, K_D, is commonly called the Doppler constant. Solving equation (16) for ρ_D yields

$$\rho_D = b \ln T_F + K \qquad (17)$$

where K is an integration which may be obtained from initial conditions.

$$K = \rho_D^\circ - b \ln T_F^\circ \qquad (18)$$

where

ρ_D° - Doppler effect at time zero

T_F° - fuel temperature at time zero

Substituting for K in equation (17) will give

$$\rho_D - \rho_D^\circ = \Delta\rho_D = b \ln(T_F/T_F^\circ) \tag{19}$$

The infinite multiplication factor now becomes

$$k_\infty = k_\infty^\circ + \rho - b \ln(T_F/T_F^\circ) \tag{20}$$

The effect of delayed neutrons is small compared to the prompt neutron effect; therefore, it may be assumed that the Doppler effect on delayed neutrons is insignificant to the overall problem. The k_∞ of equation (8) is, then, assumed to be k_∞°. With this assumption and equation (20), the neutron diffusion equation in the fuel, equation (10), may be rewritten as

$$\nabla\cdot(D_F\nabla\phi_F) + \Sigma_{aF}\phi_F[k_\infty^\circ(1-\beta)-1+\rho(1-\beta)-b(1-\beta)\ln(T_F/T_F^\circ)]$$

$$+ \sum_{i=1}^{n} \lambda_i\{\beta_i\Sigma_{aF}\int_0^t e^{-\lambda_i(t-t')}[k_\infty^\circ+\rho]\phi_F dt' + C_i^\circ e^{-\lambda_i t}\} = \frac{1}{v}\frac{\partial\phi_F}{\partial t} \tag{21}$$

To facilitate the present analysis, the number of delayed neutron groups is taken as one averaged group. So that, λ_i and β_i become $\bar{\lambda}$ and $\bar{\beta}$, respectively. This approximation should have little effect on the problem under consideration.

5. Boundary Conditions

The neutron diffusion problem involves solution of the partial differential equations (11), (12), and (21), with the following boundary, interface, and initial conditions:

1) $$\frac{\partial \phi_F}{\partial r}(0,z,t) = 0$$

2) $$\phi_F(a,z,t) = \phi_c(a,z,t)$$

3) $$D_F \frac{\partial \phi_F}{\partial r}(a,z,t) = D_c \frac{\partial \phi_c}{\partial r}(a,z,t)$$

4) $$\phi_c(b,z,t) = \phi_{co}(b,z,t)$$

5) $$D_c \frac{\partial \phi_c}{\partial r}(b,z,t) = D_{co} \frac{\partial \phi_{co}}{\partial r}(b,z,t)$$

6) $$\frac{\partial \phi_{co}}{\partial r}(c,z,t) = 0$$

7) $$\phi_F(r,\pm \tfrac{H}{2},t) = \phi_c(r,\pm \tfrac{H}{2},t) = \phi_{co}(r,\pm \tfrac{H}{2},t) = 0$$

8) $$\phi_F(\underline{r},0) = \phi_F^{\circ}(\underline{r})$$

9) $$\phi_c(\underline{r},0) = \phi_c^{\circ}(\underline{r})$$

10) $$\phi_{co}(\underline{r},0) = \phi_{co}^{\circ}(\underline{r})$$

Boundary condition 1) results from the assumed azimuthal symmetry. Interface conditions 2), 3), 4), and 5) are continuity conditions of the flux. Boundary condition 6) results from the use of an equivalent cell and basically indicates there is an equal number of neutrons transferred in and out of the cell at the outer boundary. This should be valid unless the cell is located near the outer edge of the reactor. Boundary condition 7) is an assumption that the flux is zero at the axial boundaries of the cell. Initial conditions 8), 9), and 10) are the assumed initial distributions of the neutron flux.

B. HEAT TRANSFER ANALYSIS

In this section, the principle of conservation of energy will be used to formulate the governing field equations for the heat transport in each of the three regions. A simple heat conduction model with convection heat transfer to the coolant is used to model the heat transport problem. A gap conductance model is used to describe the heat transport across the gap at the fuel-clad interface.

1. Fuel Region

Conservation of energy within the fuel region yields the unsteady heat conduction equation with a generation term

$$\nabla\cdot(k_F(\underline{r})\nabla T_F(\underline{r},t))+\dot{q}(\underline{r},t)=\rho_F(\underline{r})C_{pF}(\underline{r})\,\frac{\partial T_F}{\partial t}\,(\underline{r},t) \qquad (22)$$

where

$k_F(\underline{r})$ - thermal conductivity of the fuel

$T_F(\underline{r},t)$ - fuel temperature

$\dot{q}(\underline{r},t)$ - nuclear energy generation per unit volume

$\rho_F(\underline{r})$ - fuel density

$C_{pF}(\underline{r})$ - fuel specific heat

As in the neutronic analysis, the vector notation is intended to include only two dimensions, (r,z). The $\underline{r}$ and t will be dropped except where needed for clarification.

The nuclear generation term may be expressed as

$$\dot{q} = e\,\Sigma_{fF}\,\phi_F \qquad (23)$$

where e - nuclear energy released per fission

As can be seen, it is through the nuclear generation term that the temperature is directly coupled to the neutron flux. This coupling and the temperature dependent Doppler reactivity feedback combine to make the coupled problem non-linear.

Substituting equation (23) into equation (22) yields the governing thermal equation for the fuel

$$\nabla \cdot (k_F \nabla T_F) + e\, \Sigma_{fF}\, \phi_F = \rho_F\, C_{pF} \frac{\partial T_F}{\partial t} \tag{24}$$

2. <u>Cladding Region</u>

Conservation of energy within the clad will yield the heat conduction equation. With nuclear generation, a relatively small amount (~5%) of the energy will be released in the cladding and the coolant. However, in this analysis it is assumed that the total energy release is in the fuel region. This should not create any significant error. Using this assumption, the unsteady heat conduction equation for the cladding becomes

$$\nabla \cdot (k_c \nabla T_c) = \rho_c\, C_{pc} \frac{\partial T_c}{\partial t} \tag{25}$$

3. <u>Coolant Region</u>

Once again, conservation of energy will lead to the heat conduction equation plus an additional term which

takes into consideration the coolant flow. The governing equation is

$$\nabla \cdot (k_{co} \nabla T_{co}) - V_{co} \frac{\partial}{\partial z}(\rho_{co} C_{pco} T_{co}) = \rho_{co} C_{pco} \frac{\partial T_{co}}{\partial t} \tag{26}$$

where V_{co} - coolant flow velocity

Equations (24), (25), and (26) are the governing equations used to model the energy transport problem.

4. Interface Conditions

The interface between the fuel and the cladding may be an actual gap with a finite distance, or the surfaces may be in intermittent contact on a microscopic scale. To model the heat transfer across this interface, a gap heat transfer coefficient is introduced. The gap coefficient must take into consideration many items (e.g., radiation heat transfer across the gap, heat transfer by solid-to-solid contact, heat conduction across a gas filled gap). The prediction of this gap coefficient is extremely complicated and beyond the scope of this work. In Ref. [6], a set of values for H_{gap} is given and the axial variation of H_{gap} in this analysis is approximately the same. A cosine curve has been fitted to the sample data to determine the gap coefficient, see Figure 3. The heat flux across the fuel-clad interface is, then,

$$q = H_{gap} (T_F - T_C) \tag{27}$$

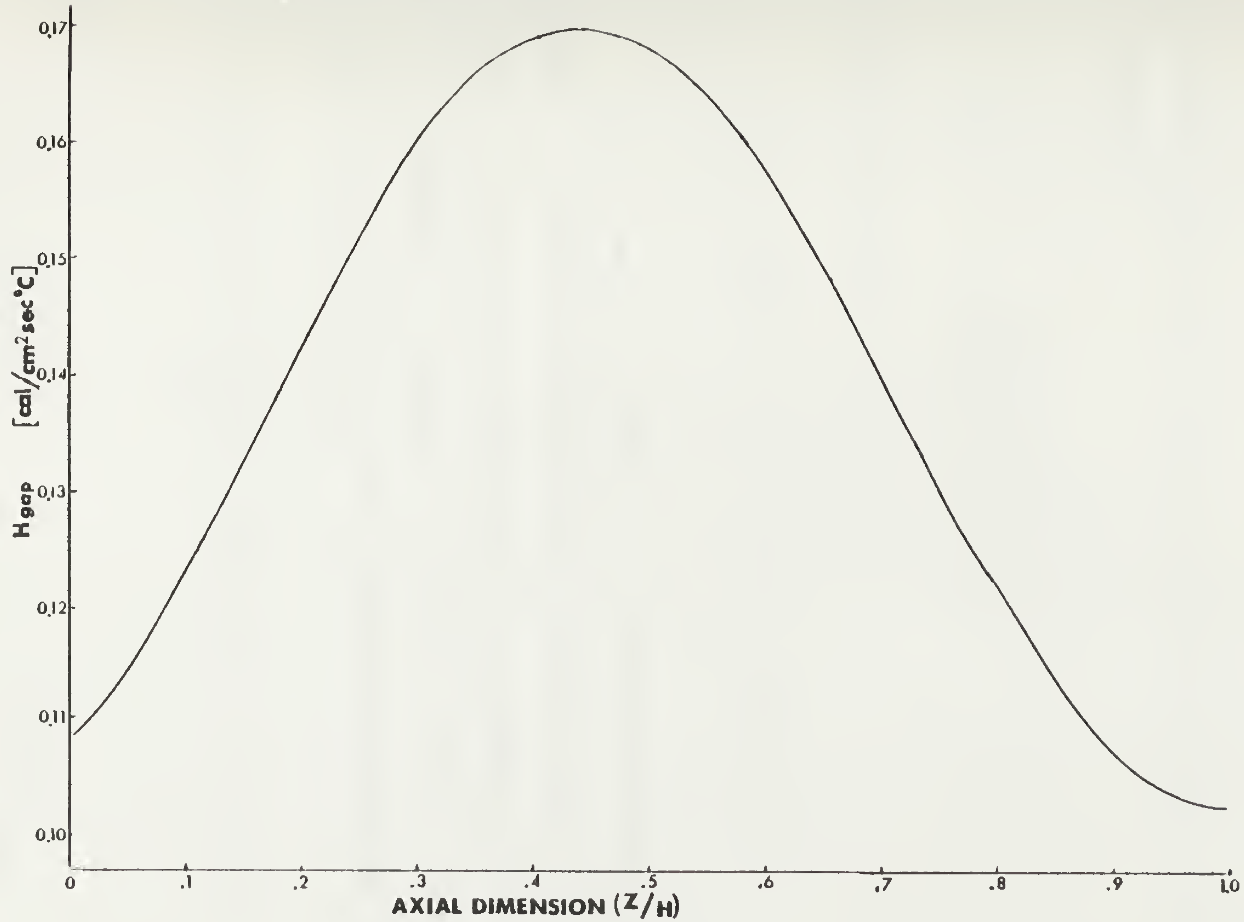


Figure 3. Gap Heat Transfer Coefficient

The heat conducted out of the fuel is governed by Fourier's equation and is equal to the heat transferred across the gap

$$q = -k_F \frac{\partial T_F}{\partial r} \qquad (28)$$

Equating equations (27) and (28) gives the fuel-clad interface condition

$$T_F(a,z,t) + \frac{k_F(a,z)}{H_{gap}(z)} \frac{\partial T_F}{\partial r}(a,z,t) = T_c(a,z,t) \qquad (29)$$

and from continuity

$$k_F(a,z) \frac{\partial T_F}{\partial r}(a,z,t) = k_c(a,z) \frac{\partial T_c}{\partial r}(a,z,t) \qquad (29a)$$

As is common practice in convection heat transfer analysis, a coolant surface heat transfer coefficient, hsurf, is used to account for the thermal resistance at the clad-coolant interface. Similar to the fuel-clad interface analysis, the clad-coolant interface conditions may be determined

$$T_c(b,z,t) + \frac{k_c(b,z)}{h_{surf}} \frac{\partial T_c}{\partial r}(b,z,t) = T_{co}(b,z,t) \qquad (30)$$

and

$$k_c(b,z) \frac{\partial T_c}{\partial r}(b,z,t) = k_{co}(b,z) \frac{\partial T_{co}}{\partial r}(b,z,t) \qquad (30a)$$

5. Boundary Conditions

The boundary and initial conditions for the heat transport problem are:

1) $\frac{\partial T_F}{\partial r}(0,z,t) = 0$

2) $T_{co}(r,-\frac{H}{2},t) = T_{PLENUM}$

3) $\frac{\partial T_{co}}{\partial z}(r,\frac{H}{2},t) = 0$

4) $\frac{\partial T_{co}}{\partial r}(c,z,t) = 0$

5) $\frac{\partial T_F}{\partial z}(r,\pm\frac{H}{2},t) = \frac{\partial T_c}{\partial z}(r,\pm\frac{H}{2},t) = 0$

6) $T_F(\underline{r},0) = T_F^{\circ}(\underline{r})$

7) $T_c(\underline{r},0) = T_c^{\circ}(\underline{r})$

8) $T_{co}(\underline{r},0) = T_{co}^{\circ}(\underline{r})$

Boundary condition 1) results from the assumed azimuthal symmetry. The coolant has been assumed to enter the flow channel at a constant temperature, T_{PLENUM}. This results in condition 2). Boundary conditions 3) and 5) result from an assumption that no heat is transferred axially from the fuel rod. Boundary condition 4) is the result of the use of the equivalent cell. Conditions 6), 7), and 8) are the assumed initial conditions.

Solution of the nonlinear coupled neutronic and energy transport problem involves the solution of the partial differential equations (11), (12), (21), (24), (25), and (26), with the appropriate boundary, interface, and initial conditions.

Several works, Refs. [4], [11], [12], have demonstrated the feasibility and success of the finite element method in solving nuclear reactor dynamics problems. The FEM

is used to reduce the partial differential equations developed in this analysis to a system of ordinary differential equations. Integration of these ordinary differential equations (ODE) yields the solution.

IV. FINITE ELEMENT FORMULATION

In this section, the basic theory underlying the finite element method is formulated. Selection of the finite elements and the shape functions for the elements are given. Some simple transformations which facilitate the integration necessary in the FEM are also presented.

A. BASIC THEORY

To obtain a numerical solution, the governing partial differential field equations are transformed into a system of ODE in finite dimensional vector space. This may be accomplished in several manners such as the finite-difference method, the variational method, or the weighted residual method. In this work, the Galerkin method (a weighted residual method) is utilized for the discretization of the spatial domain. The Galerkin procedure will be applied to each of the governing field equations. Any of these equations may be considered to be in the following form

$$\frac{\partial \psi}{\partial t} (\underline{r},t) - \ell \psi(\underline{r},t) = f\ (\underline{r},t) \qquad (31)$$

where ψ represents the unknown function, e.g., in equation (21), ψ represents ϕ_F , ℓ represents the operator for each individual equation, and f is a forcing function. In the finite element method the solution is approximated as

$$\psi(\underline{r},t) \cong \tilde{\psi}(\underline{r},t) = \sum_{i=1}^{N} N_i(\underline{r})\ \psi_i(t) = \langle N_i \rangle \{\psi_i\} \qquad (32)$$

where N - the number of degrees of freedom

N_i - the element shape function

ψ_i - unknown coordinate function

$< >$ - matrix notation for a row vector

$\{ \}$ - matrix notation for a column vector

$<N_i>$ - $<N_1 \; N_2 \; \ldots \; N_i \; \ldots \; N_N>$

$$\{N_i\} - \begin{Bmatrix} N_1 \\ N_2 \\ \vdots \\ N_i \\ \vdots \\ N_N \end{Bmatrix}$$

The residual, $R(\underline{r},t)$, is a measure of the error in this finite element approximation. The residual may be considered as

$$R = \frac{\partial\psi}{\partial t} - \ell\psi - f \qquad (33)$$

The best solution for $\tilde{\psi}$ is one which "minimizes" this residual. Various "minimums" are obtained by the weighted residual method by setting

$$\int_V W_i(\underline{r}) \; R \; dV = 0 \qquad i=1,2,\ldots N \qquad (34)$$

$W_i(\underline{r})$ - weighting functions

With the Galerkin method, the weighting functions are the shape functions defining the approximation of equation (32) (i.e., $W_i = N_i$). A noteworthy attribute of the Galerkin method is the opportunity of using an integration-by-parts

of the terms involving the second order spatial derivatives. A lower order finite element may be used than would have been possible otherwise. Once the weighting functions have been chosen, the problem becomes

$$\int_V N_i \{\frac{\partial\psi}{\partial t} - \ell\psi - f\}\, dV = 0 \qquad i=1,2,\ldots N \qquad (35)$$

The integration involved in equation (35) is carried out on the element level, taking advantage of the use of a "local" coordinate system. Once the integration is accomplished, the results are merged into a system using "global" coordinates. On the element level

$$\tilde{\psi}^e = \langle N_j\rangle^e \{\psi_j\}^e \qquad j=1,2,\ldots,N \qquad (36)$$

where the superscript e indicates the element level. Substituting $\tilde{\psi}^e$ into equation (35) and noting $\{\psi_j\}^e$ is not a function of the spatial domain yields

$$\int_V \langle N_i\rangle^e \{N_j\}^e dV^e \{\dot{\psi}_j\}^e - \int_V [\langle N_i\rangle^e \ \ell \ \{N_j\}^e - \langle N_i\rangle^e f^e] dV^e \{\psi_j\}^e = 0 \qquad (37)$$

where $\quad i,j - 1,2,\ldots N^e$

N^e - number of degrees of freedom for an element

The operator ℓ will vary depending upon which governing equation is under consideration.

B. SHAPE FUNCTIONS

The shape functions, N_i, are chosen to satisfy certain completeness and convergence criteria [13] and will depend upon the finite-element used for the spatial discretization.

Many previous works, Refs. [4], [11], and [12], utilized linear triangular shaped elements to discretize the spatial domain. This element was the first element considered. However, because the width of the cladding is very thin, elements in the cladding region would have extremely large aspect ratios (ratio of base to height) unless an extremely large number of elements in the axial direction were used. A large number of elements becomes numerically untractable. Previous experience with triangular elements had shown that large aspect ratios yield inaccurate results. Zlamal, Ref. [14], showed the error, e, when using triangular elements, is proportional to the square of the longest side, h, and inversely proportional to the sine of the smallest angle, γ

$$e \propto h^2/\sin\gamma$$

A triangular element with a large aspect ratio necessarily must have a small related angle which adversely affects the error in the FEM. Hopefully to alleviate the problem, an isoparametric, quadratic, rectangular element was selected. The aspect ratio would still be large, but experience, Ref. [15], with the use of rectangular elements indicated that a large aspect ratio is not always a detrimental factor.

The shape functions for this element are well documented, Ref. [13]. Utilizing a "local" coordinate system (See Figure 4), the shape functions may be written as

Corner nodes $\quad i = 1,3,5,7$

$$N_i = \frac{1}{4}(1+\xi_o)(1+\eta_o)(\xi_o+\eta_o-1) \tag{38}$$

Midside nodes

$$N_i = \frac{1}{2}(1-\xi^2)(1+\eta_o) \quad i=2,6 \tag{38a}$$

$$N_i = \frac{1}{2}(1+\xi_o)(1-\eta^2) \quad i=4,8 \tag{38b}$$

where

$$\xi_o = \xi\xi_i$$

$$\eta_o = \eta\eta_i$$

These normalized shape functions are shown in Figure 5.

The local and global coordinates are related by the following

$$r = \langle N_i \rangle^e \; \{r_i\}^e \tag{39}$$

and

$$z = \langle N_i \rangle^e \; \{z_i\}^e \tag{39a}$$

C. COORDINATE TRANSFORMATIONS

When using a local coordinate system, some simple transformations facilitate the integrations required by equation (37). In cylindrical coordinates with azimuthal symmetry

$$dV = 2\pi r dr dz \tag{40}$$

The derivative terms may be transformed by the following

$$\begin{Bmatrix} \dfrac{\partial N_i}{\partial r} \\ \dfrac{\partial N_i}{\partial z} \end{Bmatrix} = [J]^{-1} \begin{Bmatrix} \dfrac{\partial N_i}{\partial \xi} \\ \dfrac{\partial N_i}{\partial \eta} \end{Bmatrix} \tag{41}$$

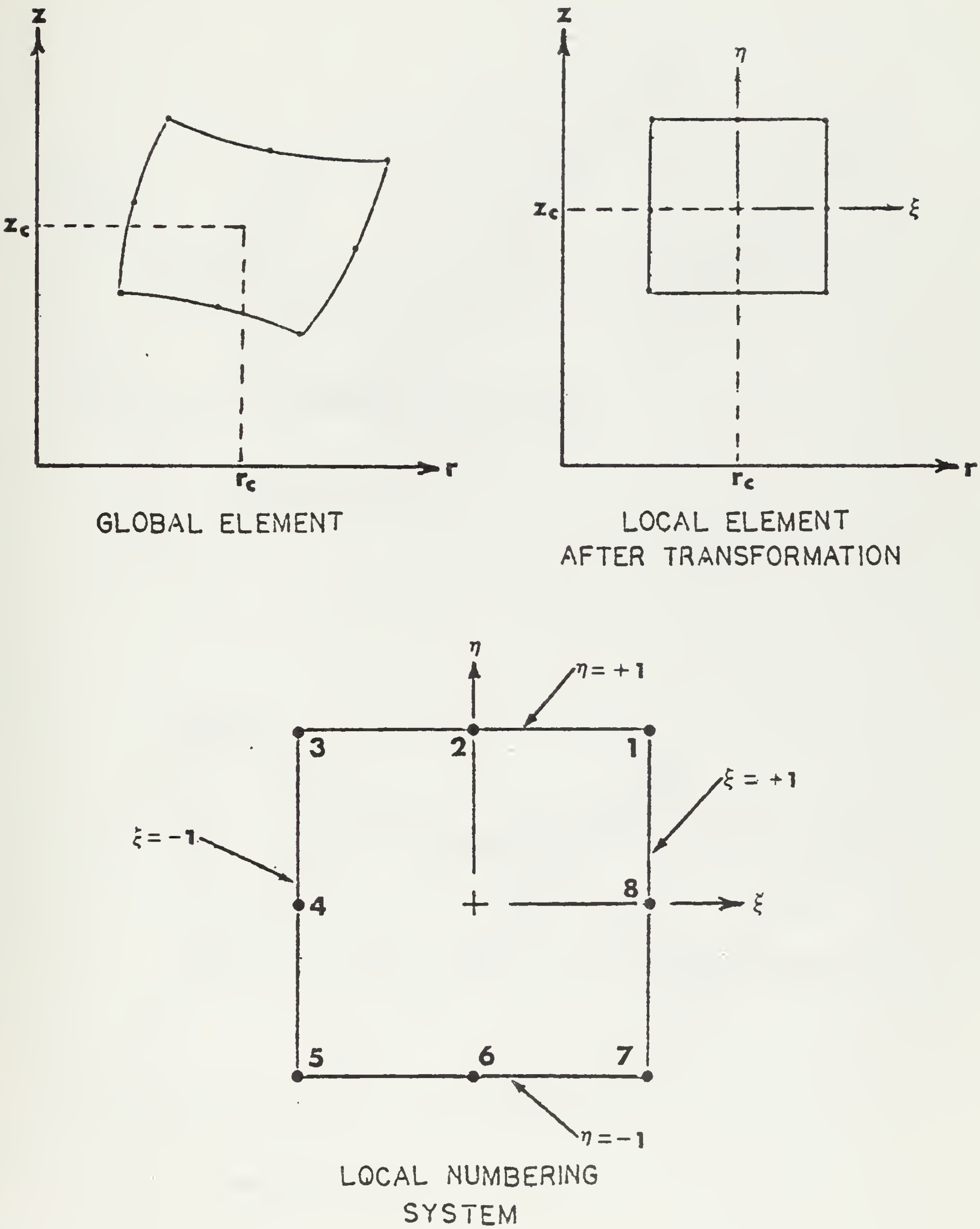
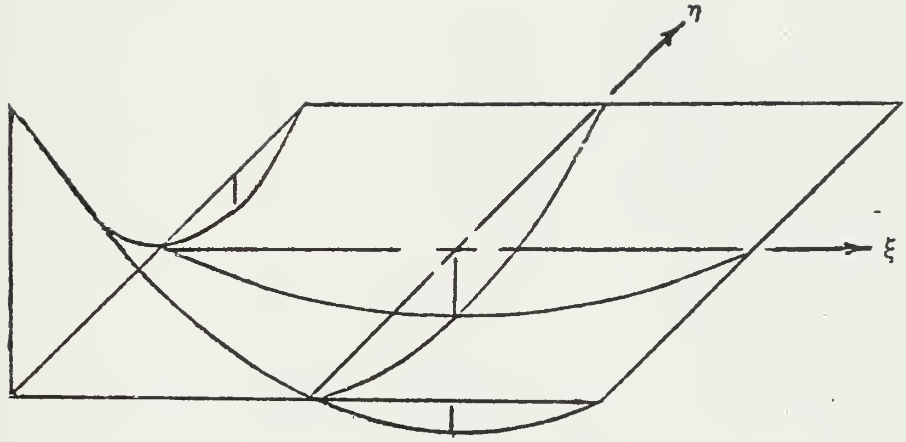
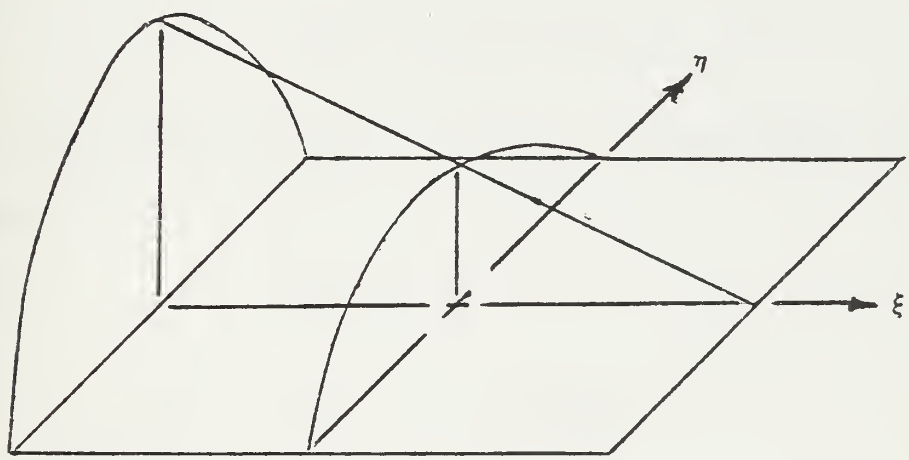


Figure 4. Element Transformation



CORNER NODE



MIDSIDE NODE

Figure 5. Normalized Shape Functions

where $[J]^{-1}$ is the inverse of the 2x2 Jacobian matrix defined in Appendix A. As shown in Appendix A, this inverse can be easily shown to be (for this problem)

$$[J]^{-1} = \begin{bmatrix} J_{11}^{*} & J_{12}^{*} \\ J_{21}^{*} & J_{22}^{*} \end{bmatrix} \tag{42}$$

where

$J_{11}^{*} = 2/\Delta r$, $J_{12}^{*} = 0$

$J_{21}^{*} = 0$, $J_{22}^{*} = 2/\Delta Z$

Δr - radial length of the element

ΔZ - axial length of the element

Elements of area transform as

$$drdz = \det[J]\ d\xi d\eta \tag{43}$$

For this particular problem, $\det[J]$ may be shown to be (Appendix A)

$$\det[J] = \frac{A^e}{4} \tag{44}$$

where A^e - area of the element

Elements of axial length become

$$dz = \frac{L^e}{2} \tag{45}$$

where L^e - axial length of the element

Utilization of these transformations makes the integrations required in equation (37) amenable to integration by numerical Gaussian quadrature.

V. APPLICATION OF FEM TO GOVERNING FIELD EQUATIONS

In this chapter, equation (37) is applied to the governing field equations previously derived. The element matrices for each of the operators are developed so that the discretization of the spatial domain may be accomplished. The integrations required by the application of equation (37) are performed numerically using Gaussian quadrature.

A. GAUSSIAN QUADRATURE

Prior to the application of equation (37), it is appropriate to discuss briefly the procedure used for the numerical integration. The product Gaussian quadrature formula is [16]

$$I_A = \int_{-1}^{1}\int_{-1}^{1} g(\xi,\eta)\,d\eta\,d\xi = \sum_{i=1}^{m}\sum_{j=1}^{m} W_i W_j\, g(\xi_i,\eta_j) \tag{46}$$

where

I_A	-	area integration
$g(\xi,\eta)$	-	any function of ξ and η
$W_{i,j}$	-	weight associated with location i or j
m	-	number of Gauss sampling points in one-dimension

The values of the weights associated with each Gauss point are given in Ref. [16]. Equation (42) may be simplified somewhat by combining the summations and weights

$$I_A = \sum_{k=1}^{m^2} W_k\, g(\xi_k,\eta_k) \tag{47}$$

where k - $i \times j$

W_k - $W_i \times W_j$

For line integrations, the Gaussian quadrature formula involves only one summation

$$I_L = \int_{-1}^{1} f(\eta)d\eta = \sum_{i=1}^{m} W_i \; f(\eta_i) \tag{48}$$

B. NEUTRONIC FIELD EQUATIONS

The discretization of the spatial domain by the finite element method is accomplished by applying equation (37) to the governing field equations, using

$$\phi_k = \langle N_j \rangle^e \; \{\psi_{kj}\}^e \qquad \begin{array}{l} j=1,2,\ldots,8 \\ k=F,c,co \end{array} \tag{49}$$

1. Fuel Region

The governing equation for the fuel region, equation (20) after applying equation (37) becomes

$$2\pi \int_r \int_z \frac{N_i}{v} \frac{\partial \phi_F}{\partial t} \, rdrdz - 2\pi \int_r \int_z N_i \{ \frac{1}{r} \frac{\partial}{\partial r}(rD_F \frac{\partial \phi_F}{\partial r})$$

$$+ \frac{\partial}{\partial z}(D_F \frac{\partial \phi_F}{\partial z}) + \Sigma_{aF}\phi_F(1-\beta)[k_\infty^\circ + \rho - b\ln(T_F/T_F^\circ)]$$

$$+ \bar{\lambda}\bar{\beta}\Sigma_{aF} \int_0^t e^{-\lambda_i(t-t')}(k_\infty^\circ + \rho)\phi_F dt' + \bar{\lambda}C^\circ e^{-\lambda t}\} rdrdz = 0$$

(50)

The second order terms in equation (50) may be reduced to a first order term by application of Green's Theorem or equivalently integration-by-parts (See Appendix B). Dividing through by 2π and reducing the second order terms yields

$$\int_z rN_i D_F \frac{\partial\phi_F}{\partial r}\,dz + \int_r rN_i D_F \frac{\partial\phi_F}{\partial z}\,dr + \int_r\int_z \frac{N_i}{v}\frac{\partial\phi_F}{\partial t}\,rdrdz$$

$$+\iint_{r\,z}\Big\{D_F\Big[\frac{\partial N_i}{\partial r}\frac{\partial\phi_F}{\partial r} + \frac{\partial N_i}{\partial z}\frac{\partial\phi_F}{\partial z}\Big] - N_i\Sigma_{aF}\phi_F(1-\beta)[k_\infty^\circ+\rho-b\ln(T_F/F_{Fo})]$$

$$-N_i\bar{\lambda}\bar{\beta}\Sigma_{aF}\int_0^t e^{-\lambda(t-t')}(k_\infty^\circ+\rho)\phi_F dt' - N_i\bar{\lambda}C^\circ e^{-\bar{\lambda}t}\Big\}\,rdrdz = 0 \quad (51)$$

From continuity and boundary conditions the line integrals are zero. Now using the approximate functions of equation (49) and noting that $\{\psi_F\}^e$ is not a function of space, equation (51) may be written as

$$\frac{1}{v}\iint_{rz}\{N_i\}^e\langle N_j\rangle^e\,rdrdz\{\dot{\psi}_F\}^e + \iint_{rz}D_F[\{N_{i,r}\}^e\langle N_{j,r}\rangle^e + \{N_{i,z}\}^e\langle N_{j,z}\rangle^e]rdrdz\{\psi_F\}^e$$

$$-\iint_{rz}\Sigma_{aF}(1-\beta)[k_\infty^\circ+\rho-b\ln(T_F/T_F^\circ)]\{N_i\}^e\langle N_j\rangle^e rdrdz\{\psi_F\}^e - \bar{\lambda}\bar{\beta}\Sigma_{aF}\int_0^t e^{-\bar{\lambda}(t-t')}(k_\infty^\circ+\rho)dt'$$

$$\iint_{rz}\{N_i\}\langle N_j\rangle rdrdz\{\psi_F\}^e - \bar{\lambda}C^\circ e^{-\bar{\lambda}t}\iint_{rz}\{N_i\}rdrdz = 0 \quad (52)$$

The integrations may not be easily carried out with a shift to local coordinates and with use of the previously derived transformations. Rearranging equation (52) and assuming the properties are constant for each time step gives

$$D_F \int_{-1}^{1}\int_{-1}^{1} [\{N_{i,\xi} J_{11}^{*}\}<J_{11}^{*} N_{j,\xi}>+\{N_{i,\eta} J_{22}^{*}\}<J_{22}^{*} N_{j,\eta}>] r\det[J]d\xi d\eta\{\psi_F\}^e$$

$$-\Sigma_{aF}(1-\beta)(k_\infty^\circ+\rho) \int_{-1}^{1}\int_{-1}^{1} \{N_i\}<N_j> r\det[J]d\xi d\eta\{\psi_F\}^e$$

$$-\Sigma_{aF}(1-\beta)b \int_{-1}^{1}\int_{-1}^{1} \ln \frac{T_F}{T_F^\circ} \{N_i\}<N_j> r\det[J]d\xi d\eta \{\psi_F\}^e$$

$$-\bar{\lambda}\bar{\beta}\Sigma_{aF}\, f(t) \int_{-1}^{1}\int_{-1}^{1} \{N_i\}<N_j> r\det[J]d\xi d\eta\{\psi_F\}^e - \bar{\lambda}C^\circ e^{-\bar{\lambda}t} \int_{-1}^{1}\int_{-1}^{1} \{N_i\} r\det[J]d\xi d\eta$$

$$+\frac{1}{v}\int_{-1}^{1}\int_{-1}^{1} \{N_i\}<N_j> r\det[J]d\xi d\eta \{\dot{\psi}_F\}^e = 0 \qquad (53)$$

Since the element chosen has eight degrees of freedom (nodal points), the discretized matrices which result from the integration of equation (53) will be 8x8 matrices and the forcing function will be an 8x1 vector at the element level. Defining the matrices as

$$\int_{-1}^{1}\int_{-1}^{1} [\{N_{i,\xi} J_{11}^{*}\}<J_{11}^{*} N_{j,\xi}>+\{N_{i,\eta} J_{22}^{*}\}<J_{22}^{*} N_{j,\eta}>] r\det[J]\, d\xi d\eta$$

$$=[H12_{ij}]_{8x8} = \frac{A^e}{4}\sum_{k=1}^{m^2} [(N_{i,\xi})_k (N_{j,\xi})_k J_{11}^{*2}$$

$$+(N_{i,\eta})_k (N_{j,\eta})_k\, J_{22}^{*}]\, r_k W_k \qquad (54)$$

where $$W_k = W_i W_j \quad \text{and} \quad r_k = \sum_{i=1}^{8} r_i (N_i)_k$$

$$\int_{-1}^{1}\int_{-1}^{1} \{N_i\}<N_j> r\det[J]d\xi d\eta = [H3_{ij}]_{8x8} = \frac{A^e}{4}\sum_{k=1}^{m^2} (N_i)_k (N_j)_k r_k W_k \tag{55}$$

$$\int_{-1}^{1}\int_{-1}^{1} \{N_i\} r\det[J]d\xi d\eta = \{F_i\}_{8x1} = \frac{A^e}{4}\sum_{k=1}^{m^2} (N_i)_k r_k W_k \tag{56}$$

$$\int_{-1}^{1}\int_{-1}^{1} \ln(T_F/T_D)\{N_i\}<N_j> r\det[J]d\xi d\eta = [H4_{ij}]_{8x8}$$

$$= \frac{A^e}{4}\sum_{k=1}^{m^2} \ln(T_F/T_{F\circ})_k (N_i)_k (N_j)_k r_k W_k \tag{57}$$

To carry out the summation of equation (57), the temperature T_F must be known; however, the temperature is exactly what is being sought. To alleviate this problem, a linearization is used.

In the solution technique, the temperature is predicted for the next time step. It is this temperature which is used for the determination of matrix H4.

Equation (53) simplifies to

$$\{D_F[H12_{ij}]-[\Sigma_{aF}(1-\beta)(k_\infty^\circ+\rho)+\bar{\lambda}\bar{\beta}\Sigma_{aF}f(t)][H3_{ij}]+\Sigma_{aF}(1-\beta)b[H4_{ij}]\}\{\psi_F\}^e$$

$$-\bar{\lambda}C^\circ c^{-\bar{\lambda}t}\{F_i\}^e + \frac{1}{v}[H3_{ij}]\{\dot{\psi}_F\} = 0 \tag{58}$$

The function $f(t)$ is evaluated by summing the values at each time step using the trapezoid rule for numerical integration.

$$f(t) = e^{-\bar{\lambda}t} \int_0^t e^{-\bar{\lambda}t'}[k_\infty^\circ+\rho]dt' \qquad (59)$$

Defining

$$I_i[g(t)] = \frac{1}{2} h_i\{g(t_i)+ g(t_{i-1})\} \qquad (60)$$

$$g(t) = e^{\bar{\lambda}t}(k_\infty^\circ+\rho)$$

h_i - time step taken

$g_o = 0$

The function may be expressed as

$$f(t) = e^{-\lambda t}\sum_{i=1}^{S} I_i\ [g(t)] \qquad (61)$$

S - number of time steps

2. Clad Region

Following the same procedure as with the fuel equation, the discretized form of equation (11) becomes

$$\{D_c[H12_{ij}] + \Sigma_{ac}[H3_{ij}]\}\{\psi_c\}^e + \frac{1}{v}[H3_{ij}]\{\dot{\psi}_c\}^e = 0 \qquad (62)$$

3. Coolant Region

The governing equation for the coolant region, equation (12), may be discretized into the form

$$\{D_{co}[H12_{ij}] + \Sigma_{aco}[H3_{ij}]\}\{\psi_{co}\}^e + \frac{1}{v}[H3_{ij}]\{\dot{\psi}_{co}\}^e = 0 \qquad (63)$$

Once the governing equations have been discretized at the element level, they are combined into a system of equations at the global level. On the global level the governing equation for neutron transport takes the form of

$$[H]_{nxn}\{\psi\}_{nx1} + [P]_{nxn}\{\dot{\psi}\}_{nx1} + \{F\}_{nx1} = 0 \qquad (64)$$

where $[H]$, $[P]$, and $[F]$ represent the system matrices and n is the number of nodal points used in the discretization. There are, then, n simultaneous ordinary differential equations used to describe the neutron transport problem.

C. HEAT TRANSPORT FIELD EQUATIONS

The spatial domain for the heat transport problem is discretized in the same manner as the domain for the neutronics problem. The same element matrices previously defined are valid. Let

$$T_k = \langle N_j\rangle^e\{\tau_{kj}\}^e \qquad j=1,2,\ldots,8 \qquad (65)$$
$$k=F,c,co$$

1. Fuel Region

The governing field equation for the fuel, equation (24), is discretized by applying equation (37). Using an integration by part to lower the order of the second order terms allows equation (24) to be written as

$$\int_r [N_i k_F \frac{\partial T_F}{\partial z}\Big|_o^z \, rdrdz + \int_z [N_i k_F \frac{\partial T_F}{\partial r}\Big|_o^a \, dz - \int_r\int_z \{k_F[\frac{\partial N_i}{\partial r}\frac{\partial T_F}{\partial r} + \frac{\partial N_i}{\partial z}\frac{\partial T_F}{\partial z}]$$
$$-e\Sigma_{fF}N_i\phi_F + N_i\rho_F C_{pF}\frac{\partial T_F}{\partial t}\} \, rdrdz = 0 \qquad (66)$$

From continuity considerations the line integrals are zero except along the boundaries of a region. It is assumed that no heat is transferred from the cell in the axial direction (boundary condition 5); therefore, the first line integral of equation (66) is zero. In the neutron diffusion problem there was continuity of flux at the interfaces so that the line integrals were zero; however, the heat transfer at the interfaces is affected by the gap and film conductances. The fuel-clad interface condition, equation (29), may be rewritten as

$$-k_F \frac{\partial T_F}{\partial r} = H_{gap}(T_F - T_c) = -k_c \frac{\partial T_c}{\partial r} \tag{67}$$

Substituting equation (67) into (66), dividing by minus one and utilizing equation (65) yields

$$\int_z [rH_{gap}\{N_i\}^e<N_j>^e|_o^a dz\{\tau_F\}^e - \int_z [rH_{gap}\{N_i\}^e<N_j>^e|_o^a dz\{\tau_c\}^{e*}$$

$$+ \int_r \int_z k_F[\{N_{i,r}\}^e<N_{j,r}>^e + \{N_{i,z}\}^e<N_{j,z}>^e]rdrdz\{\tau_F\}^e$$

$$- \int_r \int_z e\Sigma_{fF}\{N_i\}^e<N_j>^e rdrdz\{\psi_F\}^e$$

$$+ \int_r \int_z \rho_F C_{pF} \{N_i\}^e<N_j>^e rdrdz \{\dot{\tau}_F\}^e = 0 \tag{68}$$

e* - element across the interface

Transforming to local coordinates and integrating by Gaussian quadrature yields the same element matrices as given for the neutron flux, equations (54) and (55), except for the line integrals between regions. It should be noted that the line integrals exist only on the interfaces, along which there is a discontinuity of temperature. For the fuel equation the interface corresponds to the local coordinate $\xi=1$. Define

$$\int_{-1}^{1} \{N_i\}<N_j>d\eta = [k1_{ij}]_{8x8} = \frac{L^e}{2} \sum_{k=1}^{m^2} (N_i)_k(N_j)_k W_k \qquad (69)$$

where the N's are evaluated at $\xi=1$. Many of the terms of K1 will be zero since only the nodes on the $\xi=1$ boundary will have shape functions which are non-zero. It is through K1 that the temperatures for each region are coupled together. Equation (68) may now be written as

$$r_a H_{gap}\{[K1]\{\tau_F\}^e-[K1]\{\tau_c\}^{e*}\}+k_F[H12]\{\tau_F\}^e$$

$$-e\Sigma_{fF}[H3]\{\psi_F\}^e + \rho_F C_{pF}[H3]\{\dot{\tau}_F\} = 0 \qquad (70)$$

In obtaining this equation, it was assumed that material properties for each nodal point were constant at each time step. Perhaps a better assumption would have been to assume an average value for the properties of each element. The difference should not be significant, and the assumed constant nodal properties were numerically more tractable.

2. Clad Region

Applying equation (37) to the governing field equation for the clad, equation (25) gives the discretized form of the equation. Assuming no heat transfer in the axial direction on the boundaries (Boundary condition 5), equation (25) becomes

$$- \int_z [N_i r k_c \frac{\partial T_c}{\partial r}|_a^b \, dz + \int_r \int_z k_c[\frac{\partial N_i}{\partial r} \frac{\partial T_c}{\partial r} + \frac{\partial N_i}{\partial z} \frac{\partial T_c}{\partial z}] \, rdrdz$$

$$+ \int_r \int_z \rho_c \, C_{pc} \, N_i \frac{\partial T_c}{\partial t} \, rdrdz = 0 \qquad (71)$$

In the cladding, there are two interfaces along which the line integral of equation (71) is not zero, along the fuel-clad interface and along the clad-coolant interface. For the fuel-clad interface, equation (61) applies. For the clad-coolant interface, the interface condition, equation (30), may be rewritten as

$$k_c \frac{\partial T_c}{\partial r} = \text{hsurf} \; (T_c - T_{co}) = -k_{co} \frac{\partial T_{co}}{\partial r} \qquad (72)$$

Along the fuel-clad interface, the local coordinate corresponds to $\xi=-1$. Define the new element matrix

$$\int_{-1}^{1} \int_{-1}^{1} \{N_i\}^e \langle N_j \rangle^e d\eta = [K2_{ij}]_{8x8} = \frac{L^e}{2} \sum_{k=1}^{m^2} (N_i)_k \, (N_j)_k \, W_k$$

where the N's are evaluated at $\xi=-1$. As with K1, K2 will have many zero values because the shape functions are evaluated at $\xi=-1$.

Along the clad-coolant interface, the local coordinate corresponds to $\xi=1$ and the K1 matrix is appropriate.

Substituting equations (67) and (72), equation (71) becomes

$$\int_z [rh_{surf} N_i (T_c - T_{co})|_b \, dz - \int_z [rH_{gap} N_i (T_F - T_c)|_a \, dz$$

$$+ \int_r \int_z k_c [\frac{\partial N_i}{\partial r} \frac{\partial T_c}{\partial r} + \frac{\partial N_i}{\partial z} \frac{\partial T_c}{\partial z}] \, rdrdz$$

$$+ \int_r \int_z \rho_c \, C_{pc} \, N_i \frac{\partial T_c}{\partial t} \, rdrdz = 0 \qquad (73)$$

The governing equation for the clad region may now be written as

$$r_a H_{gap} \{[K2]\{\tau_c\}^e - [K2]\{\tau_F\}^{e^*}\} + r_b h_{surf} \{[K1]\{\tau_c\}^e - [K1]\{\tau_{co}\}^{e^*}\}$$

$$+ k_c [H12]\{\tau_F\}^e + \rho_c \, C_{pc} [H3]\{\dot{\tau}_c\} = 0 \qquad (74)$$

The line integrals of equation (73) affect only nodes which are on one of the boundaries; therefore, the nodal inputs into K1 and K2 are zero unless the node is on one of the boundaries.

3. Coolant Region

The field equation governing the coolant may be discretized in the same manner as above. After applying the Galerkin method and performing an integration by parts

on the second order terms, equation (25) becomes

$$-\int_z [rN_i k_{co} \frac{\partial T}{\partial r}|_b^c \, dz + \int_r \int_z k_{co}[\frac{\partial N_i}{\partial r} \frac{\partial T_{co}}{\partial r} + \frac{\partial N_i}{\partial z} \frac{\partial T_{co}}{\partial z}] \, rdrdz$$

$$+ \int_r \int_z V_{co} \, \rho_{co} \, C_{pco} \, N_i \frac{\partial T_{co}}{\partial z} \, rdrdz$$

$$+ \int_r \int_z \rho_{co} \, C_{pco} \, N_i \frac{\partial T_{co}}{\partial t} \, rdrdz = 0 \qquad (75)$$

The line integral, when evaluated at c, is zero (boundary condition 4). When evaluated at b, or correspondingly at ξ=-1, equation (72) is valid and K2 matrix is appropriate. All the terms of equation (75) have been defined except the flow term. Define

$$\int_{-1}^{1} \int_{-1}^{1} \{N_i\}^e <N_{i,\eta}>^e r \det[J] d\xi d\eta = [H5_{ij}]_{8x8}$$

$$= \frac{A^e}{4} \sum_{k=1}^{m^2} (N_i)_k \, (N_{i,\eta})_k \, r_k W_k \qquad (76)$$

Transforming to local coordinates and integrating reduces equation (75) to

$$r_b h_{surf}\{[K2]\{\tau_{co}\}^e - [K2]\{\tau_c\}^{e*}\} + k_{co}[H12]\{\tau_{co}\}^e$$

$$+ V_{co}\rho_{co}C_{pco}[H5]\{\tau_{co}\}^e + \rho_{co}C_{pco}[H3]\{\dot{\tau}_{co}\}^e = 0 \qquad (77)$$

Now that the governing equations have been defined for each region on the element level, equations (70), (74), and (77), they may be assembled into a system equation on the global level. The equation will be in the general form of

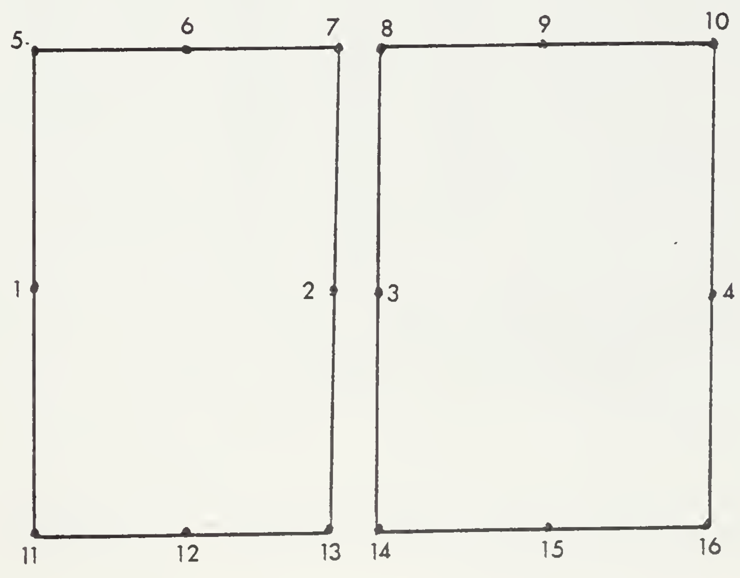
$$[K]_{nxn}\{\tau\}_{nx1} + [M]_{nxn}\{\psi\}_{nx1} + [G]_{nxn}\{\dot{\tau}\}_{nx1} = 0 \tag{78}$$

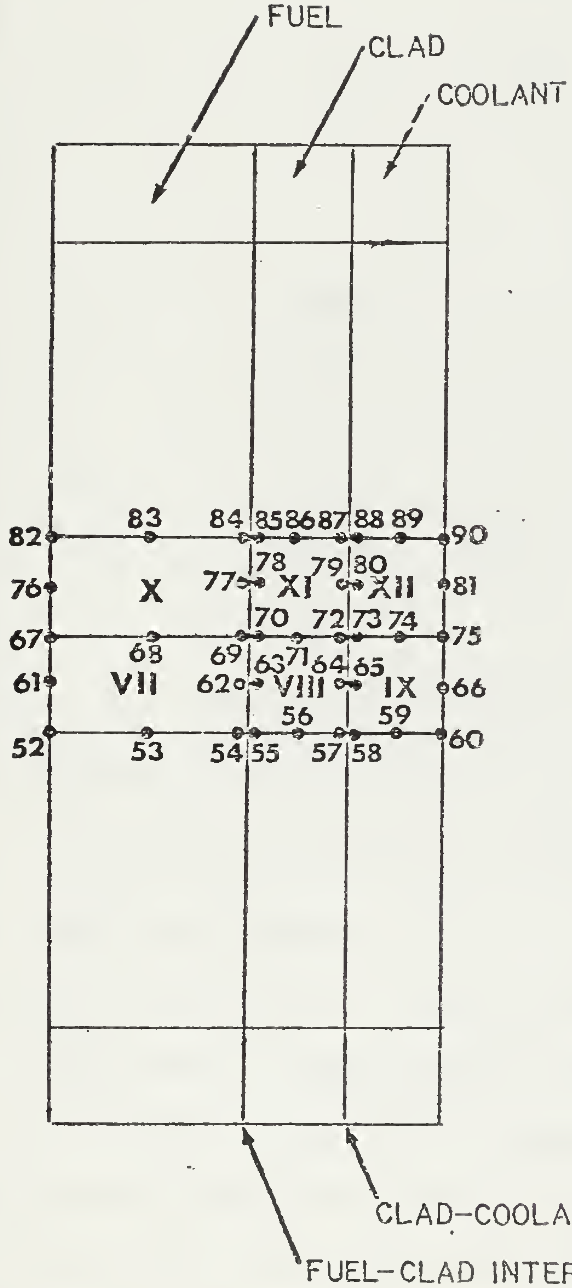
D. DISCRETIZATION OF THE SPATIAL DOMAIN

Prior to the numerical solution of the governing equations, equations (64) and (78), the spatial domain must be divided into a number of elements. For this work the domain was subdivided as shown in figure 5.

Since there is a discontinuity of temperatures at the interfaces, as described by equations (29) and (30), a noval application of the FEM method was necessary. The common practice for handling these "flux" type boundary conditions is to define a constant reference temperature, T_∞, as when working with a convection heat transfer problem [17], or to define a known function, as when working with a fracture mechanics problem [18]. In either case the reference condition was known. The novel application here lies in the use of a different field equation to describe the reference temperature, e.g., the clad equation (74) is the reference condition for heat transfer from the fuel across the gap interface.

The discontinuity of temperature at the interface necessitated another novel application of the FEM. Since there is a temperature drop along each interface, a single node there is not adequate. In the discretization of the domain, two nodes were used for each interface point (for example, points 62 and 63 in figure 6). This allows the temperature drop due to the gap and film conductances to be taken into consideration. Since two nodes are used, the governing equations for each region are not directly coupled together. The coupling of the regions is accomplished by the "flux" boundary or interface conditions since it is assumed that any heat flux leaving a region enters the adjacent region. Consider a typical set of elements on an interface





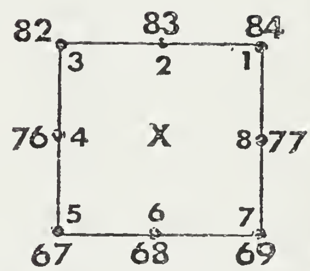


Figure 6. Finite Element Discretization

The coupling terms K1 and K2 may be combined into a system K matrix which shows the coupling. The K matrix for the simple set shown is

$$\begin{array}{cc}
 & \begin{array}{ccccccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & \cdots & 13 & 14 & 15 & 16 \end{array} \\
\begin{array}{c} 1\\2\\3\\4\\5\\6\\7\\8\\ \vdots \\13\\14\\15\\16 \end{array} &
\left[\begin{array}{ccccccccccccc}
 & & & & & & & & & & & & \\
 & a & -a & & & & c & -c & & c & -c & & \\
 & -a & a & & & & -c & c & & -c & c & & \\
 & & & & & & & & & & & & \\
 & & & & & & & & & & & & \\
 & & & & & & & & & & & & \\
 & c & -c & & & & b & -b & & d & -d & & \\
 & -c & c & & & & -b & b & & -d & d & & \\
 & & & & & & & & & & & & \\
 & c & -c & & & & d & -d & & b & -b & & \\
 & -c & c & & & & -d & d & & -b & b & & \\
 & & & & & & & & & & & & \\
 & & & & & & & & & & & &
\end{array}\right]
\end{array}
\qquad
\begin{array}{l} a = 16/15 \\ b = 4/15 \\ c = 2/15 \\ d = 1/15 \end{array}$$

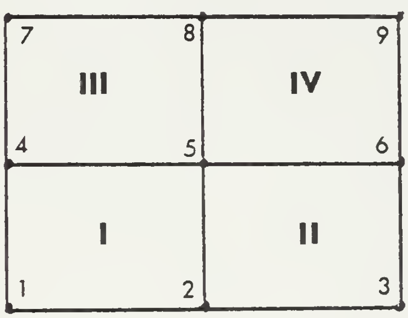
As can be seen, the nodes on the interface are coupled to the adjacent element interface nodes. For example, node 2 in element I is coupled to nodes 3, 8, and 14 in element II.

E. OPTIMUM COMPACTING SCHEME

The system matrices (H , P , etc.) are nxn matrices, where n is the number of nodal points used in the discretization of the domain. In terms of computer storage, these matrices may become excessively large if they are stored as nxn. There are several techniques available to reduce this storage requirement. The most common method is the banded storage scheme, whereby only the banded portion of the matrices are stored. With judicious numbering of the nodes,

considerable savings may be realized. However, it is not the optimum storage scheme [8].

Since the shape functions, N_i, for the $k^{\underline{th}}$ nodal equation are nonzero over only the element containing k, the system matrices are not only banded but sparse as well. The sparseness is due to the non-consecutive numbering of the nodes surrounding the $k^{\underline{th}}$ node. The optimum compact storage (OCS) scheme compacts the matrices by storing only the non-zero elements of the matrices. The implementation of the OCS scheme requires two additional integer arrays, say JA and NAME. The NAME array identifies the nodal points which contribute to each nodal equation. The JA array acts as a pointer to indicate where the nodal equation starts in NAME. Consider the following simple 2x2 system with nodes as indicated



The NAME array starts with node 1 and identifies the nodes which contribute to node 1 (i.e., 5, 4, and 2). The NAME array would, then, give the nodes contributing to node 2 and so forth, so that

```
            1234 . 5678910 .11 121314 . ... .   JC
   <NAME> = <1542 : 254163 : 3 6 5 2  : ... : 9856>
and
             1 2  3
   <JA>   = <1 5 11 ...>
```

The algorithm to assemble the element matrices into a compact storage vector is straightforward and represents a significant savings in computer storage [8]. The system matrices are stored as a vector rather than a two-dimensional array. For example, the value which would be stored in position (1,5) of the nxn array is stored in position 2 of the system vector.

VI. NUMERICAL SOLUTION

This section contains a brief description of possible solution techniques in addition to the solution technique chosen. Computer subroutines necessary to implement the technique are also described.

A. SELECTION OF METHOD

The numerical solution of the system of implicit ordinary differential equations, equations (64) and (78), may be accomplished by any of a number of different techniques such as Houbolt's method, Crank-Nicolson's method, Gear's method, or implicit Gear's method. It was not the objective of this analysis to determine which of the numerical solution schemes is the most efficient. Each method has its advantages and disadvantages. The Crank-Nicolson method is a single-step, implicit equation solver and, therefore, does not require storage of previous time solutions. When analyzing neutronic problems, the system of equations which arises is commonly very stiff (i.e., a rapid change in flux over a short period of time). The Crank-Nicolson method has, in a past work [8], demonstrated difficulty in tracking these stiff systems. Gear's method was specifically developed for stiff systems and can handle the problem very well. However, Gear's method is a multi-step, predictor-corrector method requiring storage of previous time solutions. In addition to this disadvantage, Gear's method requires the

transformation of the developed implicit O.D.E.'s into an explicit system of O.D.E.'s. After this transformation is done, the system matrices are no longer sparse or banded, thus eliminating the use of the optimum compacting scheme. In an effort to overcome these difficulties, Gear's method was modified, Ref. [9], to treat the implicit system of equations as well as to allow use of the optimum compacting scheme. A previous work, Ref. [8], has shown that the implicit Gear's method is particularly attractive in solving the type problem developed in this analysis. Therefore, the implicit Gear's method is used for the solution of the system of O.D.E.'s arising in this analysis.

No attempt will be made here to give the mathematics involved in developing the implicit Gear's method. Reference [9] may be consulted if details are desired. A listing of the computer program developed will be given in the Computer Program section. In order to utilize the implicit Gear's method, several user supplied subroutines must be developed: 1) DIFFUN, 2) JACMAT, and 3) NUITSL.

B. USER SUPPLIED SUBROUTINES TO IMPLEMENT THE IMPLICIT GEAR'S METHOD

1. DIFFUN

Subroutine DIFFUN evaluates equations (64) and (78) for a given time and for given values of ψ, $\dot{\psi}$, τ and $\dot{\tau}$. Since at each nodal point, i, there is a solution for the flux and for the temperature, the solution was set equal to DYI and DYII, respectively. In addition to having flux and

temperature at each nodal point, there are also three different regions in the domain which have different governing equations. An integer array, ITYPE, was developed to indicate for each nodal point whether it was: 0) a fuel node not in an interface element, 1) a fuel node in an interface element, 2) a cladding node or, 3) a coolant node. Using ITYPE, the computer program is directed to a different section depending upon the type of node being considered. After all the nodes have been considered, boundary conditions are established by changing DYI and DYII for the appropriate boundary nodes. Since in this analysis there is continuity of flux at the interfaces, special considerations must be given to these nodes. At the fuel-clad interface, the value of DYI for the clad node was set to the value of the flux at that node minus the value of the flux at the adjacent node (i.e., $DYI_i = \psi_i - \psi_{i-1}$). Similarly, at the clad-coolant interface, the value of DYI for the coolant node was set to the value of the flux at that node minus the value of the flux at the adjacent node. During the solution of the problem, DYI is driven toward zero, which in the limit forces ψ_i to equal ψ_{i-1}. This is the desired continuity result.

2. JACMAT

Subroutine JACMAT, evaluates the Jacobian matrix (for Gear's method) at the given time and for the current values of the dependent variables. The Jacobian for an equation of the type,

$$F(y, \dot{y}, t) = 0 \qquad (79)$$

may be represented as [19],

$$J = \left[\frac{\partial F}{\partial y} - \frac{\alpha_o}{\beta_o h} \frac{\partial F}{\partial \dot{y}}\right] \tag{80}$$

where α_o and β_o are coefficients from Gear's method and h is the time step. Using the notation of DIFFUN, let DYI and DYII represent equations (64) and (78), respectively. The Jacobian matrix may, then, be written as (J is called PW in JACMAT.)

$$PW = \left[\frac{\partial DYI}{\partial \psi} - \frac{\alpha_o}{\beta_o h} \frac{\partial DYI}{\partial \dot{\psi}} , \frac{\partial DYII}{\partial \tau} - \frac{\alpha_o}{\beta_o h} \frac{\partial DYII}{\partial \dot{\tau}}\right] \tag{81}$$

It is the form of equation (81) which is programmed in JACMAT. As in DIFFUN, the integer array ITYPE is used to indicate the appropriate section of the program to be utilized. The problem boundary conditions must also be accounted for in JACMAT. In DIFFUN, the value of DYI or DYII was set to zero for constant boundary conditions (i.e., zero). This cannot be done in JACMAT since a division by zero would occur. For a constant boundary condition at the ith node, the value of PW is set to one for the diagonal term and zero for all other terms of the ith equation.

3. NUITSL

Subroutine NUITSL solves the system of equations for the quasi-Newton iterates. In this analysis the system is solved using a successive over-relaxation (SOR) method. In this work, the optimum amount of over-relaxation was not determined. Since no effort was made to find the optimum,

it was felt a small over-relaxation would be best. The over-relaxation factor of 0.02 was used. For small values of this factor, the SOR method approaches the Gauss-Siedel iteration technique.

VII. PROCEDURE

In this section, the method utilized to obtain a solution is described. The input data necessary to run the developed computer program will be documented.

Prior to initiating a transient overpower excursion, the steady-state conditions for the fuel cell must be known. Since the system of equations which were developed are not specifically designed to obtain a steady-state solution, the initial steady-state conditions must be part of the input data. The initial temperature distribution was obtained from the steady-state conditions given in Ref. [7].

The axial temperature distribution for the fuel centerline, fuel surface, clad, and coolant have been determined for several different fuel life cycles [7]. For this analysis, the beginning of life cycle for channel 10 was used. Although this distribution is somewhat artificial, it should be adequate for this analysis. It is the trends of the results which are considered important. The distribution within the fuel radially is taken to vary as the square of the radial distance; then

$$T_F(r,z,0) = T_F(0,z,0)\left(1-\frac{r^2}{a^2}\right) + T_F(a,z,0)\left(\frac{r^2}{a^2}\right)$$

Within the cladding and the coolant, the initial radial temperature distribution is assumed to be constant.

The initial flux distribution is assumed to be radially constant, a flat flux assumption. In the axial direction the flux is assumed to vary as the shape of the sine function. The maximum flux, the flux at the axial center, is an input parameter. For this analysis, the maximum initial flux was taken to be 10^{14} neutron/cm^{2}sec.

To obtain a steady state flux distribution, the value of fission cross section for the fuel is varied. A trial-and-error method is used until a critical fission cross section, Σ_{fF}^{cr} , which gives a steady flux is obtained.

Once the steady-state conditions have been determined, the excess reactivity may be inserted. This starts the transient overpower excursion.

A. INPUT DATA

The first data card contains: the order of Gauss quadrature, the number of radial elements in the fuel, the number of axial elements, number of nodal points in the radial direction, and the height of the fuel rod. The next cards, one for each radial nodal point, contain the nodal radial distances. The next cards contain the fuel centerline, fuel surface, clad and coolant temperatures. There is one card for each axial node. The next card contains the maximum flux. The next four cards contain the physical parameters listed in Table I. The last input cards contain the time, end time, estimated initial time step, minimum time step, and maximum time step. A sample data deck is shown in figure 7.

TABLE I. Physical Parameters

Fuel Diffusion Coefficient (DCF)	0.93 [cm)
Doppler constant (B)	0.006
Energy release per fission (E)	7.652×10^{-12} [cal/fissions]
Fraction of delayed neutrons (BETA)	0.0064
Fraction of delayed neutrons for the ith group (BETAI)	0.0064
Decay constant for ith delayed neutron group (DCLAMI)	0.0784
Initial flux for delayed neutrons	1×10^{10} [neutron/cm^{2}sec]
Average number of neutrons released per fission (ANU)	2.44
Neutron velocity (VEL)	4.8×10^{8} [cm/sec]
Fuel absorption cross section (SIGAF)	0.088 [cm^{-1}]
Critical fuel fission cross section (SIGFF)	0.0586875 [cm^{-1}]
Blanket absorption cross section (SIGAB)	0.0 [cm^{-1}]
Blanket fission cross section (SIGFB)	0.0 [cm^{-1}]
Step reactivity input (RHOA)	Variable
Ramp reactivity input (RHOB)	Variable
Fuel density (DENF)	10.9 [gm/cm^3]
Clad diffusion coefficient (DCC)	1.1 [cm]
Clad specific heat (CPC)	0.12 [cal/gm °C]
Clad density (DENC)	8.0 [gm/cm^3]
Clad thermal conductivity (TKC)	0.0526 [cal/cm sec °C]
Clad absorption cross section (SIGAC)	0.0015 [cm^{-1}]
Coolant diffusion coefficient (DCCO)	1.55 [cm]
Coolant absorption cross section (SIGACO)	0.00004 [cm^{-1}]
Coolant flow velocity (VCO)	396.0 [cm/sec]
Surface heat transfer coefficient(HSURF)	0.7 [cal/cm^{2}sec °C]

```
0.0        2.0        1.0E-12    1.0E-20    0.1
1.55       4.0E-05 396.0         0.7
1.1        0.12       8.0        0.0526     0.0015
 4.8E08    0.086   .588800E-1    0.049      0.0        0.0        0.064    10.9
0.98       0.006   7.652E-12     0.0064     0.0064     0.0764     1.0E10   2.44
1.00E14
1420.      847.       594.       575.
1460.      853.       591        571.
                                      :
                           ONE DATA CARD PER AXIAL NODE
                                      :
1580.      755.       413.       387.
1540.      743.       407.       380.
0.365
0.3285
0.292
0.292
0.273
0.254
0.254
0.127
0.0
  3     1    22     9   33.0
```

SAMPLE DATA DECK

Figure 7. Sample Data Deck

VIII. RESULTS

When using the finite element method, one of the first considerations must be given to the convergence of the method. To determine convergence, the results for a given point are compared for different finite element discretizations. As shown in figure 8, the results are comparable but no definite claim of convergence can be made. However, for this work, it was felt that these results were adequate. It was not the object of this analysis to arrive at the "final" result; it was the trends and methods that were of interest. Since the 66-element mesh appears to give a fair approximation of the results, the 66-element mesh was used as the discretized domain.

The next item of consideration was the determination of a neutronic steady-state condition. This proved to be a very time consuming task. The fission cross section for the fuel was varied by a trial-and-error method in an attempt to find the critical cross section which would give a steady state. As may be seen in figure 9, a change in cross section of less than one percent significantly affected the state of the problem. It was felt that the critical value was between the values of 0.05875 and 0.058625. Time did not permit investigation for critical value; therefore, it was assumed that the value for the critical fission cross section was half way between the values (i.e., $\Sigma_f^{cr} = 0.0586875$).

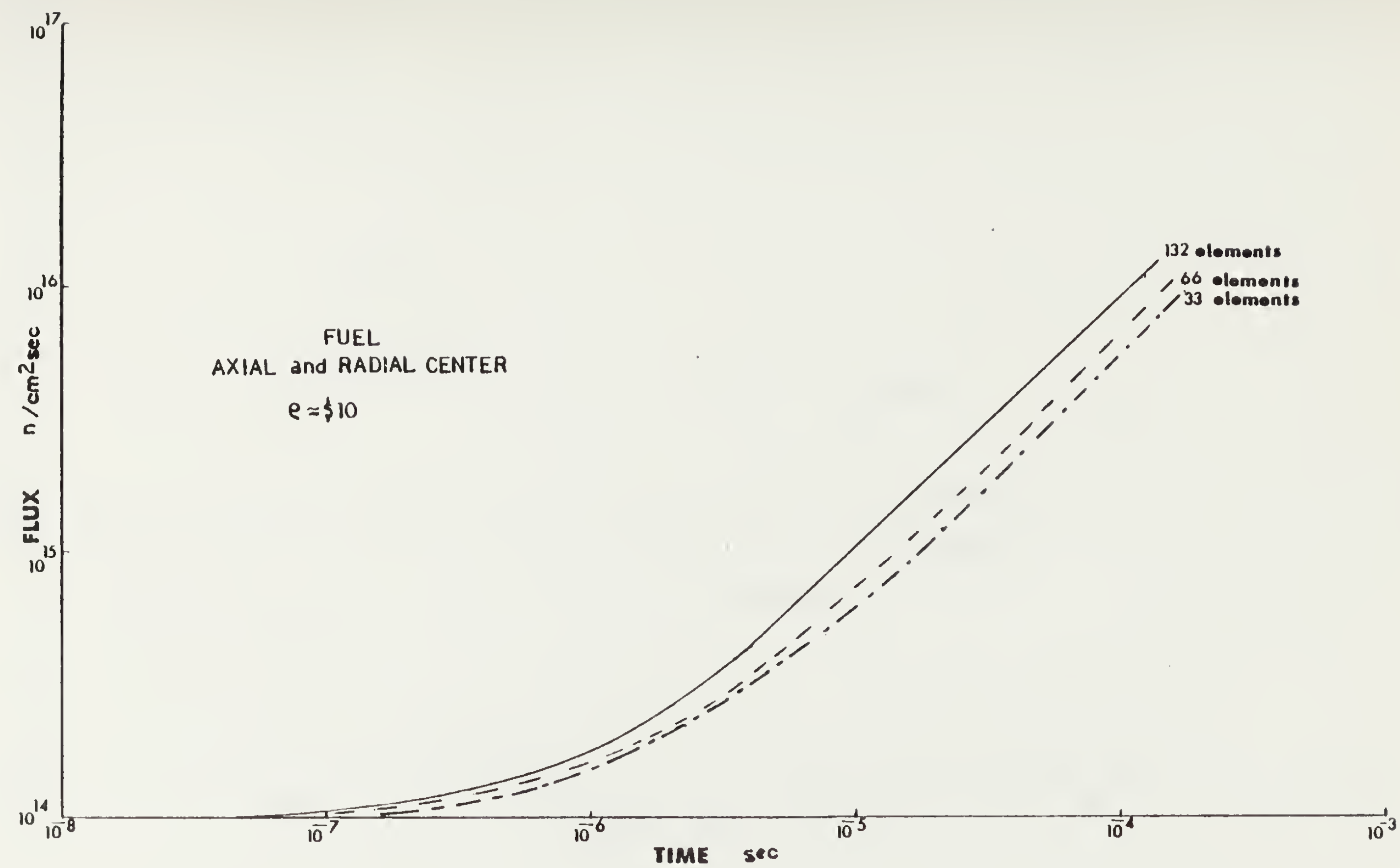


Figure 8. Convergence of Finite Element Method

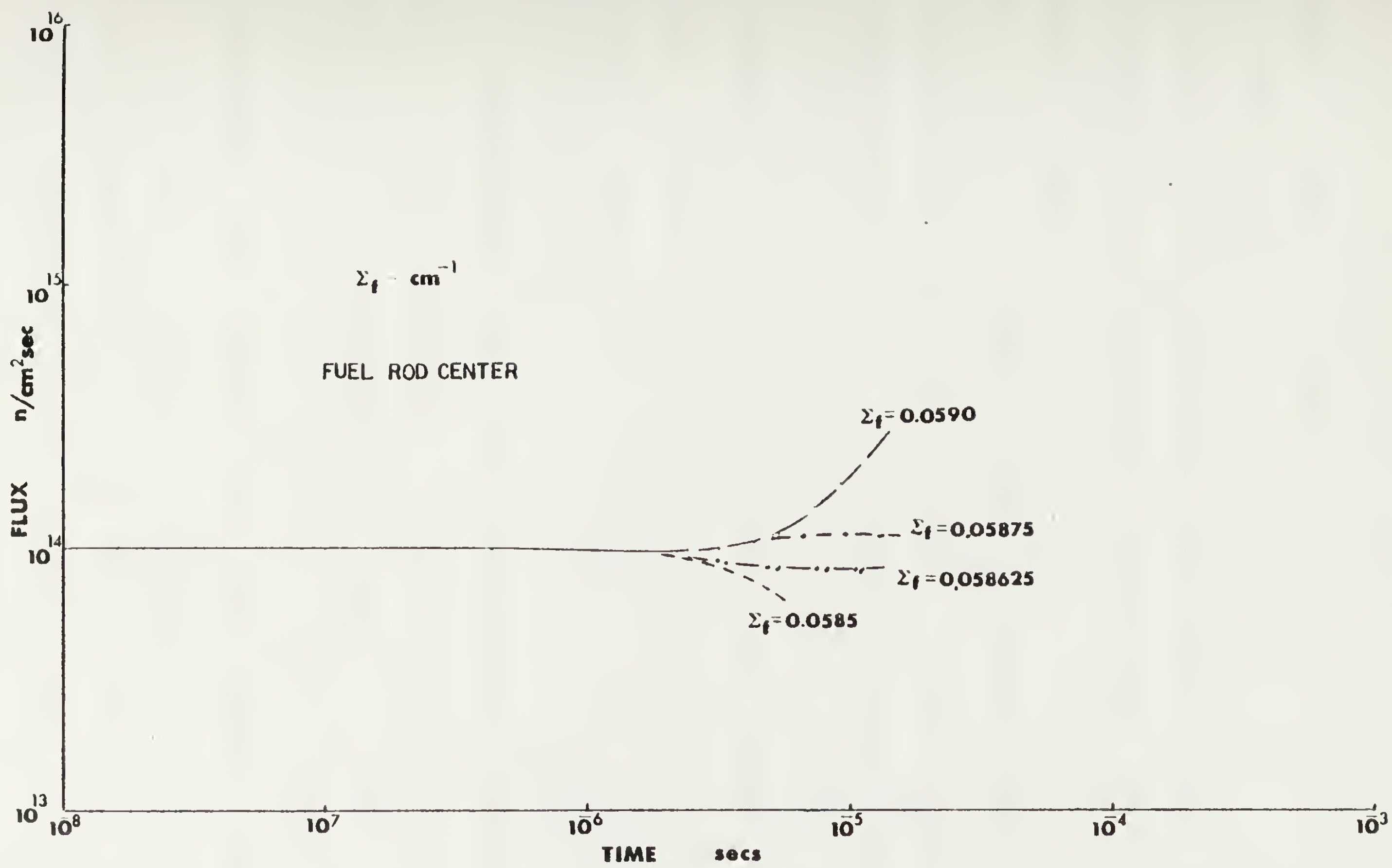


Figure 9. Determination of Critical Fission Cross Section

Even if this value is in error, which it most probably is, the net effect would only be a small decrease or increase in the proposed reactivity insertion. The reactivity insertion would, then, be an approximation of the actual reactivity of the problem.

The first test problem considered was a step increase in reactivity of approximately ten dollars. For the uranium dioxide fuel, one dollar of reactivity was taken to be 0.0064. Figure 10 shows the time history of the flux at the center of the fuel rod. Figure 11 gives the corresponding temperature profile. The temperatures were taken at the hottest point of each region at the axial center (i.e., the fuel centerline, clad inside surface, and coolant inside surface). As seen on the fuel temperature time history, the fuel rapidly reaches the fuel melting point. The model developed does not take into consideration melting of the fuel. This melting would tend to decrease the effect of the transient. The problem was allowed to continue despite this inconsistency in the mathematical model. A short time after fuel melting, the inside surface of the clad reaches its melting point. The temperature in the coolant experienced what is felt to be a numerical phenomenon. The coolant temperature decreased prior to the small rise at the end of the transient. Intuitively, this decrease does not seem to be realistic. A similar occurrence was observed while conducting sample tests on the developed computer program. Reference [20] reported the same phenomenon. It is felt that this

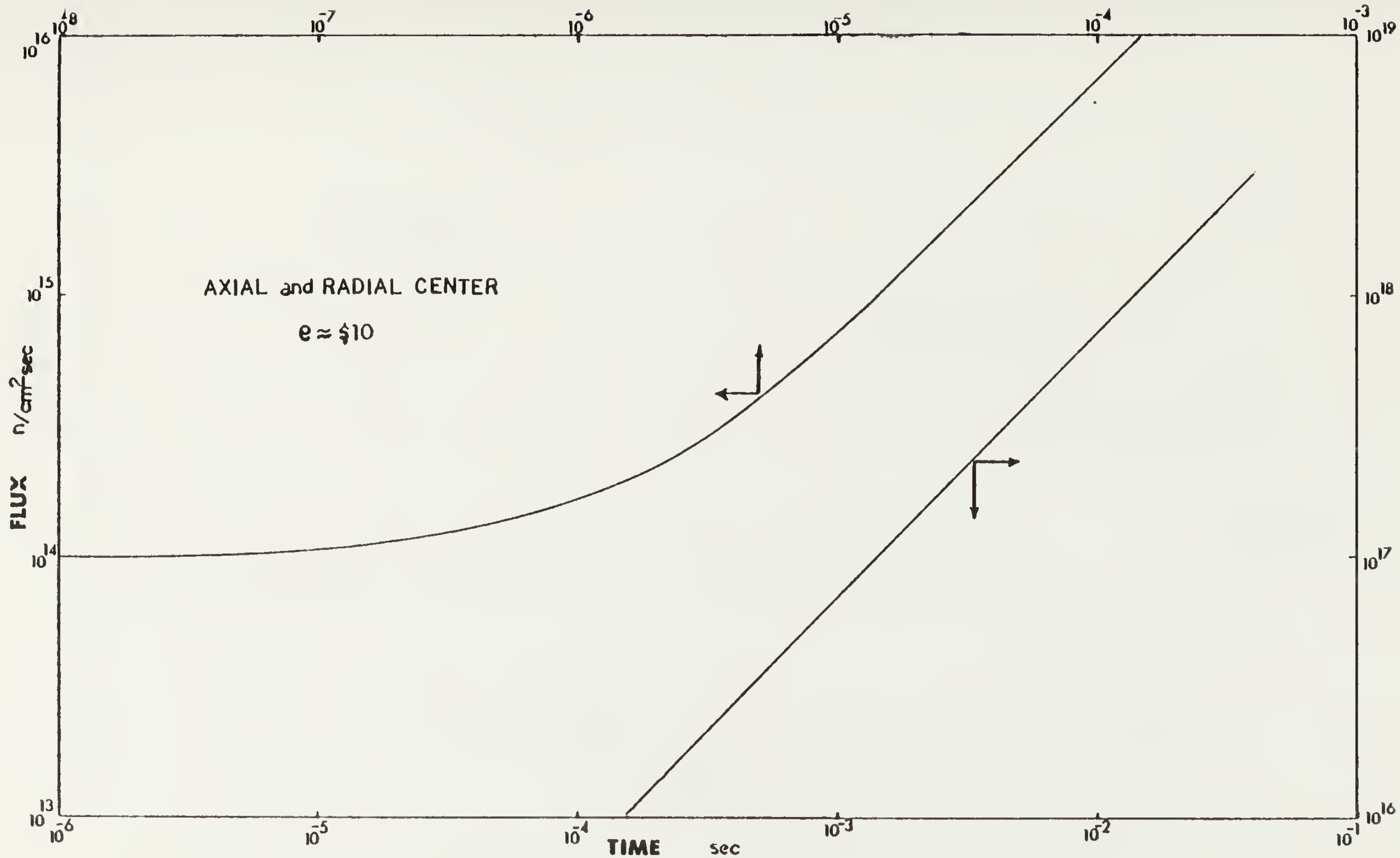


Figure 10. Flux Profile

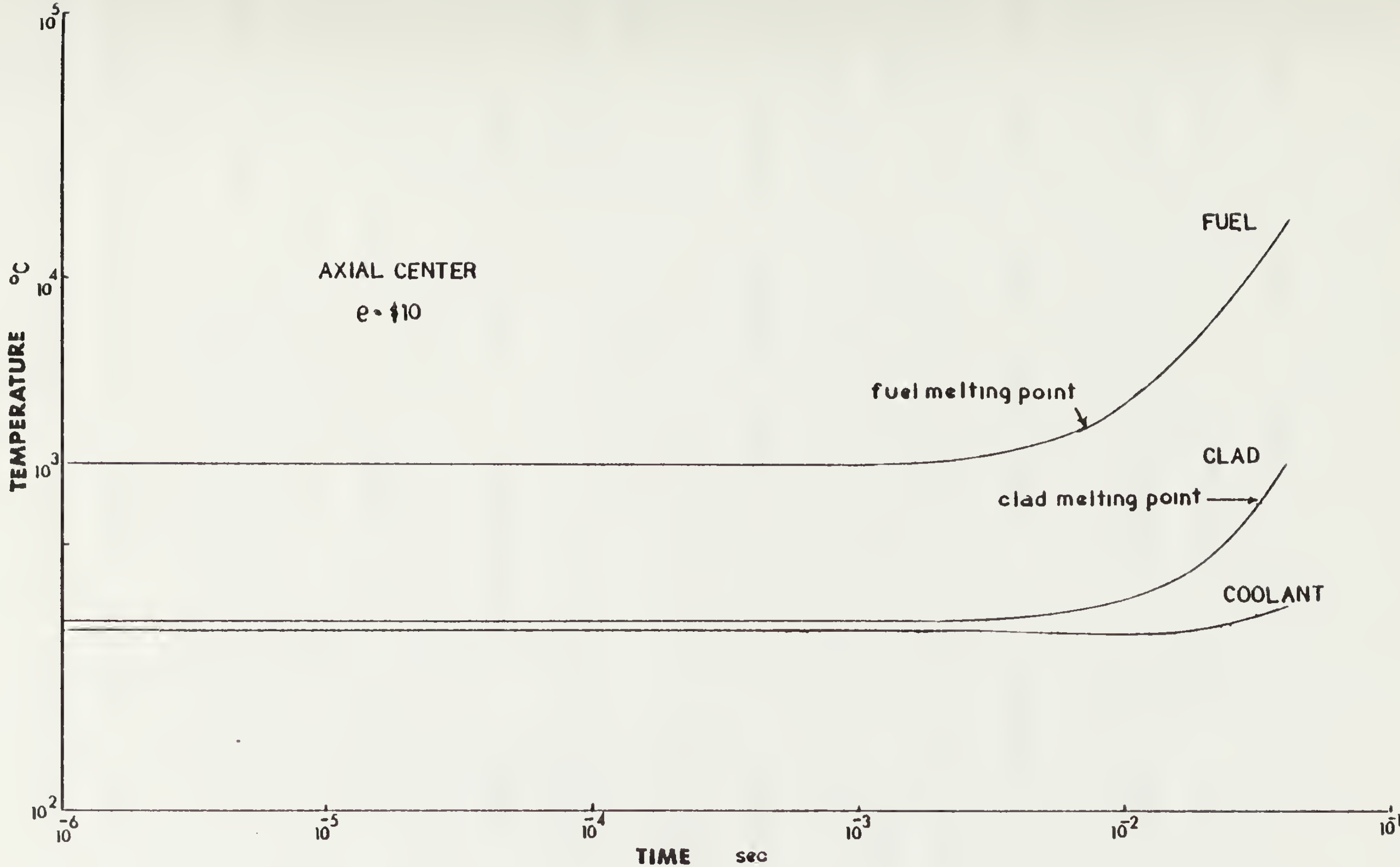


Figure 11. Temperature Profile

phenomenon is a quirk of the finite element method. The radial and axial distributions of the neutron flux are presented in figures 12 and 13 for time equal to 7.39×10^{-3} seconds. The distributions are, basically, as anticipated. The neutron flux peaks slightly before the axial center. It was expected to peak at the axial center. The radial and axial temperature distributions for the same time are presented in figures 14 and 15. As with the flux, the temperature profiles were, basically, as expected. The fuel temperature peaks slightly below the expected location, most likely in response to the peak in axial fluxes. The coolant unexpectedly drops near the outlet of the fuel rod. The finite element method characteristically has some problems on the boundaries of the domain; this may account for the drop in coolant temperature.

The temperature of the fuel and cladding do not appear to be as closely coupled as anticipated. As seen in figure 14, a significant increase in fuel temperature has resulted in a relatively small clad temperature increase. As noted in figure 11, there appears to be a time lag in temperature response for each region which may account for part of the apparent temperature disagreement. It is felt that the temperatures should be more closely related, which indicates a higher gap heat transfer coefficient should be utilized. Since values of the gap heat transfer coefficient were assumed, it is not unreasonable to believe the values used are too low.

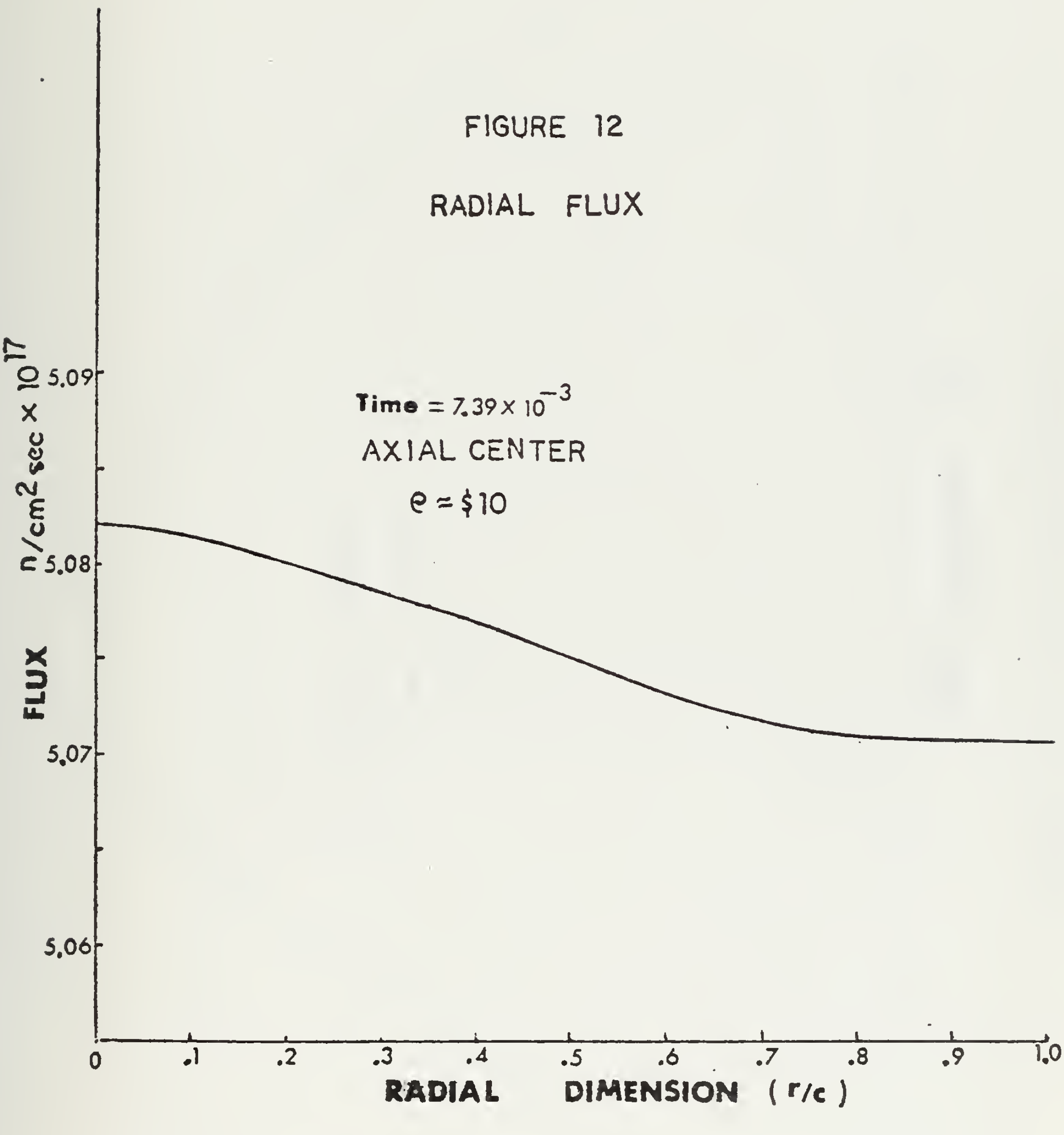


Figure 12. Radial Flux

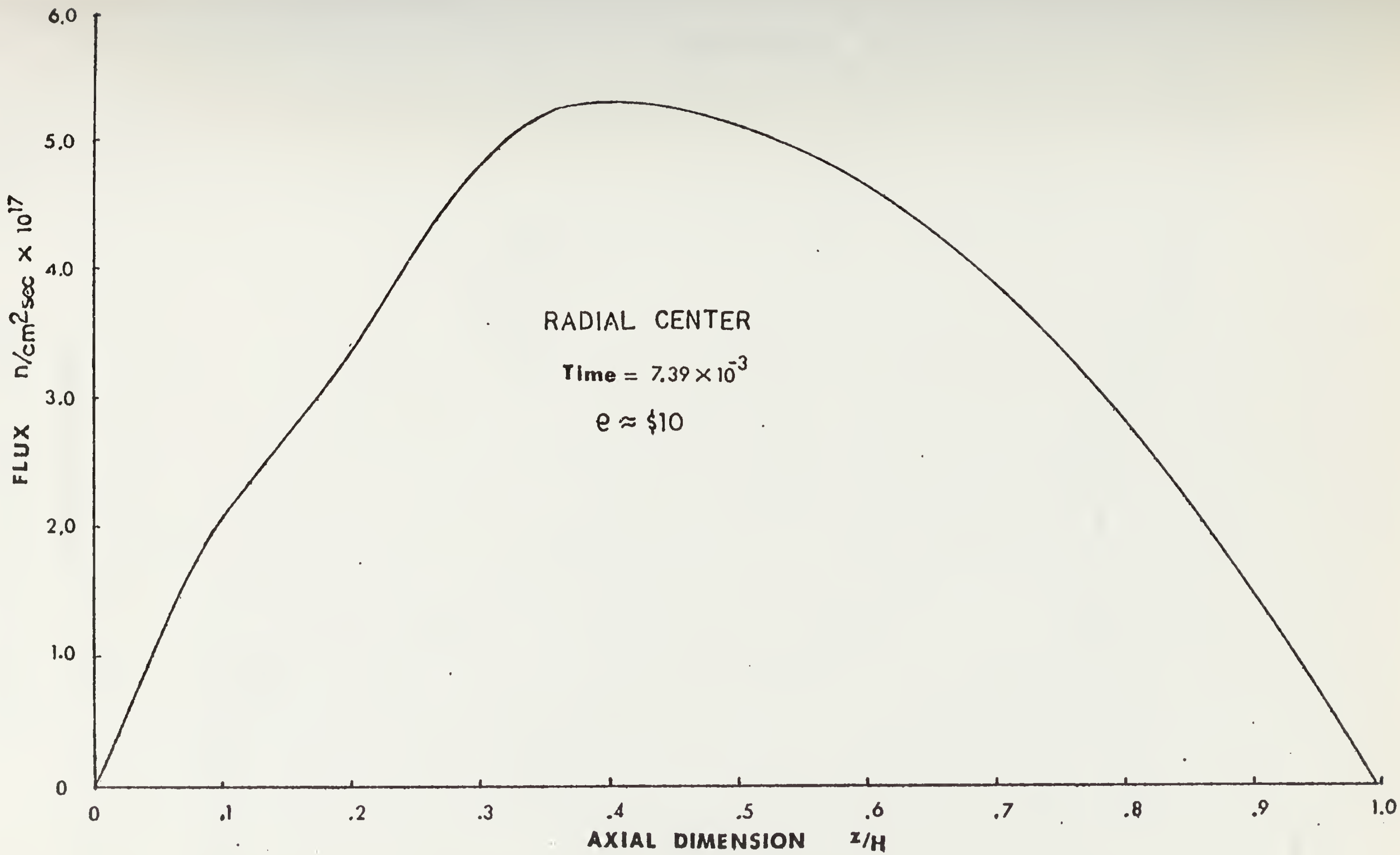


Figure 13. Axial Flux Profile

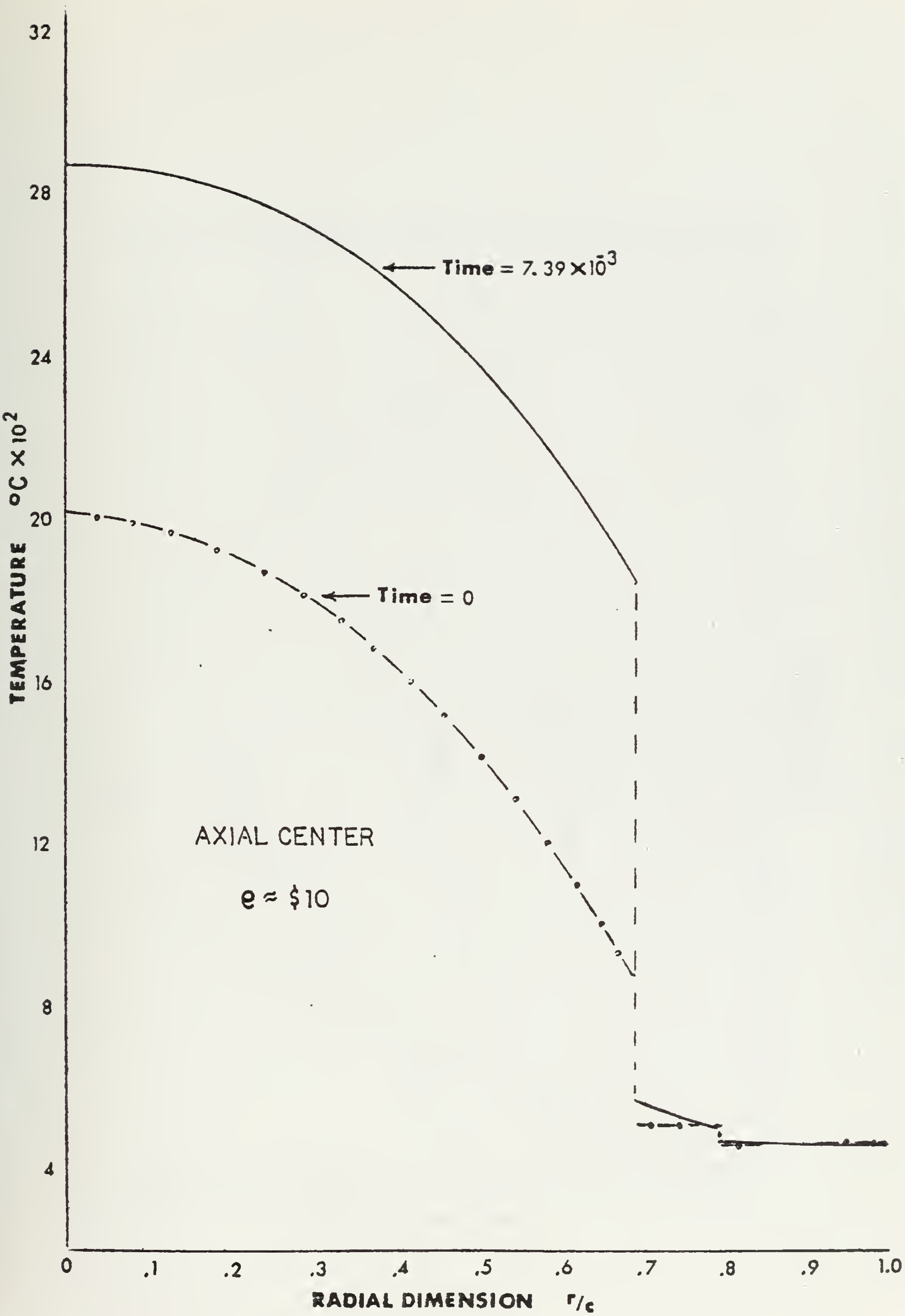


Figure 14. Radial Temperature Profile

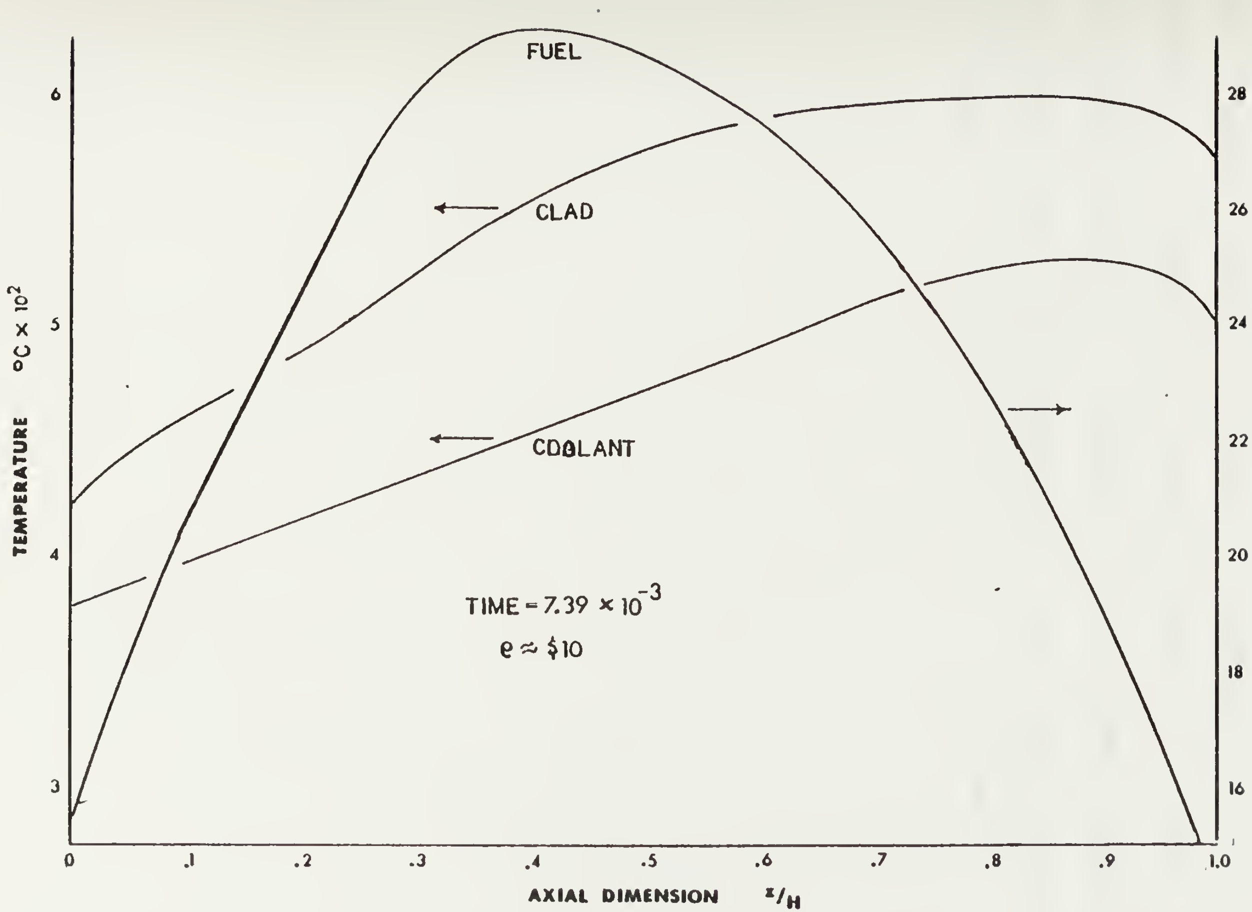


Figure 15. Axial Temperature Profile

Time did not permit investigation of other reactivity insertions. Other reactivity inputs may be investigated by students in the future.

This work does not represent a solution to the very complicated nuclear reactor problem. It does represent an application of a numerical technique which is relatively new to nuclear applications. Methods for implementing the finite element method have been discussed, and a computer code has been developed for the simplistic model considered.

IX. RECOMMENDATIONS

For the model developed, perhaps the most important item to pursue is the critical fission cross section. A better determination of this value is necessary so that the reactivity insertion is more accurately known. Different test cases for the prompt critical and prompt subcritical reactor could then be conducted.

In further developing the model, more consideration should be given the gap heat transfer coefficient. As noted in the results, the value used appears to be too small. Sample problems for different gap heat transfer coefficients would give a better indication of the values to use.

Melting of the fuel during the transient would probably be the next major improvement on the model. With relatively few changes, the model could be adapted to allow melting element by element. This, too, would be an approximation but, still, an improvement to the model. Perhaps at the same time, a simplified model to take into consideration the fuel restructuring could be implemented.

Another improvement would be to consider reactivity feedbacks in addition to the Doppler feedback. Sodium voiding and fuel rod expansion are two of the more important feedback effects to consider.

On the numerical side, probably the most important thing to do would be to run the computer program on the

"H-compiler", which optimizes the program. However, on several runs using the H-compiler, erroneous results were obtained. With sufficient time, this could be corrected to allow use of the H-compiler. The use of the H-compiler results in a savings in computer time. The present program runs on a "G-compiler" and takes excessive amounts of computer time (two to four hours per run).

In addition to this, the optimum over-relaxation factor in the implicit Gear's method could be determined by trial-and-error.

Implementation of these recommendations should enhance the analysis and lead to a more efficient computer code.

APPENDIX A

DEVELOPMENT OF TRANSFORMATIONS

The Jacobian matrix [J] may be written (8) for two dimensions as

$$[J] = \begin{bmatrix} \sum_{i=1}^{N} N_{i,\xi} r_i & \sum_{i=1}^{N} N_{i,\xi} z_i \\ \sum_{i=1}^{N} N_{i,\eta} r_i & \sum_{i=1}^{N} N_{i,\eta} z_i \end{bmatrix} . \tag{A1}$$

For a simple 2x2 matrix [A] the inverse is

$$[A] = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$[A]^{-1} = \frac{1}{\det[A]} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

Applying this fact to equation (A1) gives

$$[J]^{-1} = \frac{1}{\det[J]} \begin{bmatrix} \sum_{i=1}^{N} N_{i,\eta} z_i & -\sum_{i=1}^{N} N_{i,\xi} z_i \\ \sum_{i=1}^{N} N_{i,\eta} r_i & \sum_{i=1}^{N} N_{i,\xi} r_i \end{bmatrix} \tag{A2}$$

$$= \begin{bmatrix} J_{11}^{*} & J_{12}^{*} \\ J_{21}^{*} & J_{22}^{*} \end{bmatrix}$$

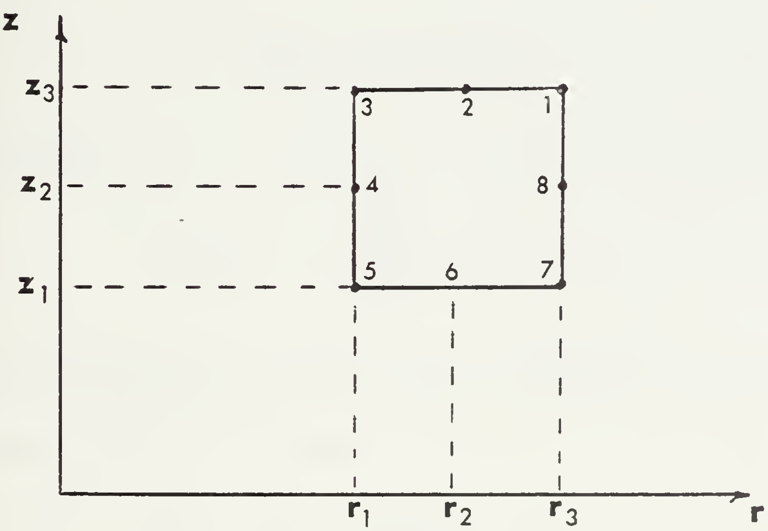
From matrix algebra

$$\det[J] = \sum_{i=1}^{N} N_{i,\xi} r_i \sum_{i=1}^{N} N_{i,\eta} z_i - \sum_{i=1}^{N} N_{i,\eta} r_i \sum_{i=1}^{N} N_{i,\xi} z_i \qquad \text{(A3)}$$

The derivatives of the shape functions may be found from equations (38).

$$\begin{aligned}
N_{1,\xi} &= \tfrac{1}{4}(1+\eta)(2\xi+\eta) & N_{1,\eta} &= \tfrac{1}{4}(1+\xi)(2\eta+\xi) \\
N_{2,\xi} &= -\xi(1+\eta) & N_{2,\eta} &= \tfrac{1}{2}(1-\xi^2) \\
N_{3,\xi} &= \tfrac{1}{4}(1+\eta)(2\xi-\eta) & N_{3,\eta} &= \tfrac{1}{4}(1-\xi)(2\eta-\xi) \\
N_{4,\xi} &= -\tfrac{1}{2}(1-\eta^2) & N_{4,\eta} &= -\eta(1-\xi) \\
N_{5,\xi} &= \tfrac{1}{4}(1-\eta)(2\xi+\eta) & N_{5,\eta} &= \tfrac{1}{4}(1-\xi)(2\eta+\xi) \\
N_{6,\xi} &= -\xi(1-\eta) & N_{6,\eta} &= -\tfrac{1}{2}(1-\xi^2) \\
N_{7,\xi} &= \tfrac{1}{4}(1-\eta)(2\xi-\eta) & N_{7,\eta} &= \tfrac{1}{4}(1+\xi)(2\eta-\xi) \\
N_{8,\xi} &= \tfrac{1}{2}(1-\eta^2) & N_{8,\eta} &= -\eta(1+\xi)
\end{aligned} \qquad \text{(A4)}$$

If one now considers an arbitrary element



with the midside nodes exactly at the midpoint (not a necessary criteria for the FEM) and substitutes into equation (A3),

the result is

$$\det[J] = \frac{(z_3-z_1)(r_3-r_1)}{4} = \frac{A^e}{4} \tag{A5}$$

Substituting into equation (A2) and using (A5) will yield

$$J_{11}^* = \frac{4}{A^e}\left[\frac{1}{2}(z_3-z_1)\right] = 2/r_3-r_1 \ , \tag{A6}$$

$$J_{12}^* = 0 \ , \tag{A7}$$

$$J_{21}^* = 0 \ , \tag{A8}$$

and

$$J_{22}^* = \frac{4}{A^e}\left[\frac{1}{2}(r_3-r_1)\right] = 2/z_3-z_1 \ . \tag{A9}$$

The inverse of the Jacobian matrix now becomes

$$[J]^{-1} = \begin{bmatrix} 2/r_3-r_1 & 0 \\ 0 & 2/z_3-z_1 \end{bmatrix} \tag{A10}$$

For integration along a line, the transformation used is

$$dz = \det[J']d\eta \tag{A11}$$

In this case

$$\det[J'] = \sum_{i=1}^{N} N_{i,\eta} z_i \tag{A12}$$

Again considering the arbitrary element and substituting (A4) into (A12) will result in

$$\det[J'] = \frac{z_3-z_1}{2} = \frac{L^e}{2} \tag{A13}$$

APPENDIX B

REDUCTION OF SECOND ORDER TERM

The second order term of the governing field equations may be reduced to first order by integration by parts. Consider, for example,

$$\int_r\int_z N_i\left[\frac{1}{r}\frac{\partial}{\partial r}\left(rD\frac{\partial\psi}{\partial r}\right) + \frac{\partial}{\partial z}\left(D\frac{\partial\psi}{\partial z}\right)\right] rdrdz \qquad \text{(B1)}$$

which may be expanded as

$$\int_r\int_z \left[N_i rD\frac{\partial^2\psi}{\partial r^2} + N_i r\frac{\partial D}{\partial r}\frac{\partial\psi}{\partial r} + N_i D\frac{\partial\psi}{\partial r} + N_i rD\frac{\partial^2\psi}{\partial z^2} + N_i r\frac{\partial D}{\partial z}\frac{\partial\psi}{\partial z}\right] drdz \qquad \text{(B2)}$$

Integrating just the second order terms by parts will yield

$$\int_r\int_z N_i D\frac{\partial^2\psi}{\partial r^2} rdrdz = \int_z \left[N_i rD\frac{\partial\psi}{\partial r}\right|_r dz - \int_r\int_z \frac{\partial\psi}{\partial r}\left[DN_i + rN_i\frac{\partial D}{\partial r} + rD\frac{\partial N_i}{\partial r}\right] drdz \qquad \text{(B3)}$$

and

$$\int_r\int_z N_i D\frac{\partial^2\psi}{\partial z^2} rdrdz = \int_r \left[N_i D\frac{\partial\psi}{\partial z}\right|_z rdr - \int_r\int_z \frac{\partial\psi}{\partial z}\left[\frac{\partial N_i}{\partial z} D + N_i\frac{\partial D}{\partial z}\right] rdrdz \qquad \text{(B4)}$$

Substituting the results of (B3) and (B4) into equation (B2) will give

$$\int_z \left[N_i rD\frac{\partial\psi}{\partial r}\right|_r dz + \int_r \left[N_i D\frac{\partial\psi}{\partial z}\right|_z rdr - \int_r\int_z D\left[\frac{\partial N_i}{\partial r}\frac{\partial\psi}{\partial r} + \frac{\partial N_i}{\partial z}\frac{\partial\psi}{\partial z}\right] rdrdz \qquad \text{(B5)}$$

APPENDIX C

LIST OF RELATIONS FOR MATERIAL THERMAL PROPERTIES

A. FUEL (UO_2)

1. Specific Heat, Ref. [19]

$$C_{pF} = [18.45+2.431\times10^{-3}T-2.272\times10^{-5}T^2]/270.07 \quad [cal/gm\ °C]$$

T - °C

2. Thermal Conductivity, Ref. [5]

$$Tk_F = [1-2.5(1-\rho_{TD})]\times[\frac{45.1}{135+T} + 4.79\times10^{-13}T^3]\times0.239 \quad [cal/cm\ sec\ °C]$$

ρ_{TD} - percent theoretical density

T - °K

B. CLAD (Stainless Steel)

Properties are assumed to be temperature independent, and average values from Ref. [5] were used for the clad properties.

C. COOLANT (Liquid Sodium)

1. Specific Heat, Ref. [5]

$$C_{p_{co}} = 0.34574-0.79226\times10^{-4}T+0.34086\times10^{-7}T^2 \quad [cal/gm\ °C]$$

T - °F

2. Density, Ref. [5]

$$\rho_{co} = [59.566-7.9504\times10^{-3}T-0.2872\times10^{-6}T^2 + 0.06035\times10^{-9}T^3] \times 0.01601 \quad [gm/cm^3]$$

T - °F

3. Thermal Conductivity, Ref. [5]

$$Tk_{co} = [54.306-1.878\times10^{-2}T+2.0914\times10^{-6}T^2] \times 4.134\times10^{-3} \quad [\text{cal/cm sec °C}]$$

T - °F

D. SURFACE HEAT-TRANSFER COEFFICIENT

$$h_{surf} = \frac{Tk_{co}}{De}\left[7.0+0.025\left(\frac{De\ V_{co}\rho_{co}C_{p_{co}}}{Tk_{co}}\right)^{0.8}\right] \quad [\text{BTU/hr ft}^2\ \text{°F}]$$

Tk_{co} - [BTU/hr ft °F]

ρ_{co} - [lbm/ft^3]

V_{co} - [ft/sec]

$C_{p_{co}}$ - [BTU/lbm °F]

De - equivalent diameter [ft]

```
C     MAIN PROGRAM CALLS THE VARIOUS SUBROUTINES
C
      IMPLICIT INTEGER*2(I-N)
      INTEGER*4 NY,NL,M,JSKF
      REAL*8 SN,DESN,DXSN,WT,GP,STJ11,STJ22,DETJ,XW,XSN
C
C     THE FOLLOWING DIMENSION STATEMENTS MUST BE
C     CHANGED FOR EACH DIFFERENT PROBLEM CONSIDERED
C
      DIMENSION BIGH12(5525),BIGH3(5525),BIGH4(5525),BIGH5(5525),
     1   BIGK(5525),NAME(5525)
      DIMENSION BIGF(350),R(350),Z(350),YIT(350),JA(350),JB(350),
     1   ITYPE(350),MAME(26,350)
      DIMENSION Y(7,700),W(28000),NELCON(11,66),NSTART(66),IBF(66)
C
C     APPROPRIATE DIMENSIONS ARE
C             BIGH12,BIGH3,BIGH4,BIGH5,BIGK,NAME-----------JC
C             BIGF,R,Z,YIT,JA,JB,ITYPE---------------------NP
C             MAME(26, BY ---------------------------------NP
C             NELCON(11, BY -------------------------------NEL
C             NSTART,IBF-----------------------------------NEL
C             Y,W AS REQUIRED BY FRANKE'S SUBROUTINES
C             JC IS THE TOTAL NUMBER OF NODES SEEN BY THE NODES
C             NP IS THE NUMBER OF NODAL POINTS
C             NEL IS THE NUMBER OF ELEMENTS
C     ALL THE REST OF THE SUBROUTINES WILL HAVE THEIR DIMENSIONS
C     CHANGED THOUGH THE CALLING STATEMENTS SO THAT THE ABOVE
C     DIMENSIONS ARE THE ONLY ONES WHICH REQUIRE CHANGING
C
      DIMENSION DR(13)
      DIMENSION H12(8,8),H3(8,8), F(8)
      DIMENSION H4(8,8)
      DIMENSION GP(5), SN(8,25), DESN(8,25), DXSN(8,25), STJ11(25),XW(5)
      DIMENSION STJ22(25), XSN(5,5), WT(25)
      DIMENSION NDUM(3)
      DIMENSION XK1(8,8), XK2(8,8)
      COMMON/CLAD/ DCC,CPC,DENC,TKC,SIGAC,HGAP
      COMMON/CONN/ NP,NEL,JC
      COMMON/COOL/DCCO,SIGACC,VCO,HSURF
      COMMON/FUEL/DCF,B,E,BETA,BETAI,CCLAMI,FFLUXC,AKINF,VEL,
     1   SIGAF,SIGFF,SIGAB,SIGFB,DENF
      COMMON/GP/ GP,NORC
      COMMON/INP/DR,HEIGHT,NPR,NEZ,NFE
      COMMON/JACINV/ STJ11,STJ22,DETJ
      COMMON/K1MAT/XK1,XK2
      COMMON /MATRIX/ H12, H3, F
      COMMON/MTRXH4/ H4
      COMMON/REACT/ RHOA,RHOB
```

```
      COMMON/SHAFUN/SN, DESN, DXSN
      COMMON/TIME/ FF,G,RHO
      COMMON /WEIGHT/ WT
   10 FORMAT (4I5,F10.5)
   11 FORMAT('0',9X,'JSKF = ',I5)
   15 FORMAT (8F10.5)
   20 FORMAT ('0',8(8F15.6,/))
   25 FORMAT ('0',10X,'ORDER OF GAUSS QUADRATURE IS',I5,//,10X,'NUMBER O
     1F RADIAL FUEL ELEMENTS IS',I5,//,10X,'NUMBER OF AXIAL FUEL ELEMENT
     2S IS',I5,//,10X,'NUMBER OF RADIAL NODAL POINTS IS',I5,//,10X,
     3  'FUEL ROD HEIGHT (CM) IS',F10.5)
   30 FORMAT (F10.5)
   35 FORMAT ('0','     RADIAL NODAL COORDINATES',/,5X,13F9.4)
   40 FORMAT('0',14X,'DCF',10X,'B',11X,'E',9X,'BETA',8X,'BETAI',6X,
     1    'DCLAMI',8X,'NU',9X,'KINF')
   41 FORMAT (9X,9E12.5)
   42 FORMAT('0',12X,'FFLUXO',8X,'VEL',8X,'SIGAF',7X,'SIGFF',7X,
     1    'SIGAB',7X,'SIGFB',7X,'SIGAC',6X,'SIGACC')
   43 FORMAT('0',14X,'DCC',9X,'CPC',8X,'DENC',9X,'TKC',8X,'CCCC',
     1    10X,'VCO',8X,'HSURF',8X,'DENF')
   44 FORMAT('0',5X,'RHC = RHCA + RHCB*T',//,6X,'RHCA = ',F10.3,
     1 10X,'RHOB = ',F10.3)
C     INPUT THE ORDER OF THE GAUSS QUADRATURE ---- 3,4, OR 5
C     INPUT THE NUMBER OF RADIAL ELEMENTS IN THE FUEL ---1 OR 2
C     INPUT THE NUMBER OF AXIAL ELEMENTS
C     INPUT THE NUMBER OF NODAL POINTS IN THE RADIAL DIRECTION--2*NFE+7
C     INPUT THE HEIGHT OF THE FUEL ROD ---- CM
      READ (5,10) NORD,NFE,NEZ,NPR,HEIGHT
      WRITE (6,25) NORD,NFE,NEZ,NPR,HEIGHT
C     INPUT THE RADIAL NODAL DISTANCES
      READ (5,30)(DR(I),I = 1,NPR)
      WRITE (6,35) (DR(I),I=1,NPR)
      CALL GINNY (ITYPE,NELCON,NSTART,R,Z)
      CALL CPCOMP (IBP,JA,JB,MAME,NAME,NELCON,NSTART)
      CALL INIT (IBP,NELCON,R,Z,Y)
      CALL SHAPE
      DO 45 I = 1,JC
      BIGH12(I) = 0.0
      BIGH3(I) = 0.0
      BIGH5(I) = 0.0
      BIGK(I) =0.0
   45 CONTINUE
      DO 50 I = 1,NP
      BIGF(I) = 0.0
C
C     STORE YIT -- THE INITIAL TEMPERATURE DISTRIBUTION
C
      YIT(I) = Y(1,I+NP)
```

```
   50 CONTINUE
      CALL MATK1 (NELCON)
      DO 100 MM= 1,NEL
      CALL JACOB (MM,NELCON,R,Z)
      CALL MATH12 (MM,NELCON,R,Z)
      CALL SYSH12 (MM,BIGH12,BIGH3,BIGH5,BIGF,JA,JB,NAME,NELCON)
      IF(NELCON(11,MM).EQ. 0) GO TO 100
      CALL SYSK12 (MM,JA,JB,NAME,NELCON,R,Z,BIGK)
  100 CONTINUE
C *** INPUT FUEL PHYSICAL CONSTANTS
      READ(5,15) DCF,B,E,BETA,BETAI,DCLAMI,FFLUXO,ANU,VEL,SIGAF,SIGFF,
     1   SIGAB,SIGFB,RHOA,RHOB,DENF
      AKINF = ANU*SIGFF/SIGAF
C *** INPUT CLAD PHYSICAL CONSTANTS
      READ(5,15) DCC,CPC,DENC,TKC,SIGAC
C *** INPUT COOLANT PHYSICAL CONSTANTS
      READ(5,15) DCCO,SIGACC,VCO,HSURF
      WRITE (6,40)
      WRITE(6,41) DCF,B,E,BETA,BETAI,DCLAMI,ANU,AKINF
      WRITE (6,42)
      WRITE(6,41) FFLUXO,VEL,SIGAF,SIGFF,SIGAB,SIGFB,SIGAC,SIGACC
      WRITE (6,43)
      WRITE(6,41) DCC,CPC,DENC,TKC,DCCO,VCO,HSURF,DENF
      WRITE(6,44) RHOA, RHOB
      JSKF = 0
      NY = NP*2
      NL = 0
      M = NY
      EPS = 0.01
      READ(5,15) T,TEND,H,HMIN,HMAX
  200 CALL SDESOL (Y,YL,T,TEND,NY,NL,M,JSKF,6,1,H,HMIN,HMAX,EPS,W,
     1 BIGH12,BIGH3,BIGH5,BIGK,BIGF,IBP,ITYPE,JA,JB,NAME,NELCON,R,Z,YIT,
     2  BIGH4)
      WRITE(6,11) JSKF
      STOP
      END
```

```
      SUBROUTINE GINNY (ITYPE,NELCON,NSTART,R,Z)
C
C     SUBROUTINE GINNY DEVELOPES THE SYSTEM MESH
C     IT ALLOWS FOR ONE OR TWO RADIAL ELEMENTS IN THE FUEL
C     AND ONE ELEMENT IN THE CLAD AND COOLANT
C    NFE SETS THE NUMBER OF RADIAL FUEL ELEMENTS
C     THE GENERATOR ALLOWS FOR ANY NUMBER OF AXIAL ELEMENTS DESIRED
C     NUMBER OF AXIAL ELEMENTS SET BY NEZ
C     THE  GENERATOR WILL GIVE THE R AND Z DIMENSION FOR EACH NODE
C     IT ALSO DEVELOPES THE CONNECTIVE MATRIX
C
C     ITYPE -- AN ARRAY USED TO INDICATE THE TYPE OF NODE
C              0 -- FUEL NODE NOT IN AN INTERFACE ELEMENT
C              1 -- FUEL NODE IN AN INTERFACE ELEMENT
C              2 -- CLADDING NODE
C              3 -- COOLANT NODE
C     NELCON -- THE CONNECTIVE MATRIX
C     R -- RADIAL DIMENSION OF THE ITH NODE
C    Z -- AXIAL DIMENSION OF THE ITH NODE
C
      IMPLICIT INTEGER*2(I-N)
      DIMENSION DR(13)
      DIMENSION ITYPE(1)
      DIMENSION NELCON(11,1)
      DIMENSION NSTART(1)
      DIMENSION R(1), Z(1)
      COMMON/CONN/ NP,NEL,JC
      COMMON/INP/DR,HEIGHT,NPR,NEZ,NFE
      LL = 0
      NP = 0
      KK = 1
      M = 2*NFE + 1
      MM = M + 4
      MMM = M + 2
      NEL = NFE + 2
      NPZ = NEZ*2
      NADD =   3*NFE + 14
      EZ = NEZ
      Z1 = HEIGHT/(2.0*EZ)
      DO 100 I = 1,NPZ
      T = I
      IF (KK .EQ. 1) GO TO 80
      LL = LL + 1
      NSTART(LL) = NP + 1
   60 DO  70 J = 1,NPR
      NP = NP + 1
      Z(NP) = Z1*T
      R(NP) = DR(J)
```

```
      KK = 1
   70 CONTINUE
      GO TO 100
   80 DO 90 JJ = 1,NPR,2
      IF (JJ .EQ. MMM) GO TO 90
      NP = NP + 1
      Z(NP) = Z1*T
      R(NP) = DR(JJ)
      IF ((JJ .EQ. M) .OR. (JJ .EQ. MM)) GO TO85
      GO TO 90
   85 NP = NP + 1
      Z(NP) = Z1*T
      R(NP) = DR(JJ)
      KK = 0
   90 CONTINUE
  100 CONTINUE
      DO 125 J = 1,NPR
      NP = NP + 1
      Z(NP) = 0.0
      R(NP) = DR(J)
  125 CONTINUE
      NNEZ = NEZ - 1
      IF(NNEZ .EQ.0) GO TO 151
      DO 150 LL = 1,NNEZ
      NST = NSTART(LL)
      NOUT = NST + M + 1
      NCHA = NOUT - 2
      NEND = NST + NPR - 3
      NCOUNT = 0
      DO 150 NN = NST,NEND,2
      IF (NN .EQ. NCHA) GO TO 150
      NNN = NN
      NEL = NEL + 1
      NCOUNT = NCOUNT + 1
      IF (NN .EQ. NOUT) NNN = NN - 1
      NELCON (1,NEL) = NNN + NADD
      NELCON (2,NEL) = NELCON (1,NEL) - 1
      NELCON (3,NEL) = NELCON (2,NEL) - 1
      NELCON (4,NEL) = NNN + NPR + 1 - NCOUNT
      NELCON (5,NEL) = NNN
      NELCON (6,NEL) = NNN + 1
      NELCON (7,NEL) = NNN + 2
      NELCON (8,NEL) = NELCON(4,NEL) + 1
  150 CONTINUE
  151 NST = NSTART(1)
      NEND = NST + NPR - 3
      NOUT = NST + M + 1
      NCHA = NOUT - 2
```

```
      N = 0
      DO 153 I = NST,NEND,2
      II = 0
      IF(I .EQ. NOUT) II=1
      IF (I .EQ. NCHA) GO TO 153
      N = N +1
      NELCON(1,N) = I + 2- II
      NELCON(2,N) = I + 1- II
      NELCON(3,N) = I - II
      NELCON(4,N) = I - NFE - 4 - N - II
      IF (I .EQ. NEND) II=2
      NELCON(5,N) = NP - NPR + 2*N - 1 + II
      NELCON (6,N) = NELCON(5,N) + 1
      NELCON (7,N) = NELCON(5,N) + 2
      NELCON (8,N) = NELCON(4,N) + 1
  153 CONTINUE
      M = NFE + 2
      DO 159 I = 1,NFE
      IF(I .EQ. NFE) GO TO 156
      DO 155 J = I,NEL,M
      NELCON(11,J) = 0
  155 CONTINUE
      GO TO 159
  156 DO 158 J = I,NEL,M
      NELCON(11,J) = 1
      NELCON(11,J+1) = 2
      NELCON(11,J+2) = 3
  158 CONTINUE
  159 CONTINUE
      DO 180 J = 1,NEL
      JJ = NELCON(11,J)
      DO 180 I = 1,8
      II = NELCON(I,J)
      ITYPE(II) = JJ
  180 CONTINUE
      WRITE (6,15)
      DO 160 I = 1,NP,3
      I1 = I + 1
      IF(I1 .GT. NP) GO TO 161
      I2 = I + 2
      IF(I2 .GT. NP) GO TO 162
  160 WRITE (6,20) I,R(I),Z(I),I1,R(I1),Z(I1),I2,R(I2),Z(I2)
      GO TO 165
  161 WRITE(6,20) I,R(I), Z(I)
      GO TO 165
  162 WRITE(6,20) I,R(I), Z(I),I1, R(I1), Z(I1)
  165 WRITE (6,25)
      DO 170 J = 1,NEL
```

```
170 WRITE (6,30)  (J, (NELCON(I,J), I = 1,8))
 15 FORMAT (10X,'NODAL COORDINATES',//,10X,'NODE',4X,'R VALUE',6X,'Z V
   1ALUE',10X,'NODE',4X,'R VALUE',6X,'Z VALUE',10X,'NODE',4X,'R VALUE'
   2,6X,'Z VALUE',//)
 20 FORMAT (10X,I4,2(3X,F10.4),9X,I4,2(3X,F10.4),9X,I4,2(3X,F10.4))
 25 FORMAT (10X,'CONNECTIVITY MATRIX',//,10X,'ELEM',3X,'NOD1',3X,'NOD2
   1',3X,'NOD3',3X,'NOD4',3X,'NOD5',3X,'NOD6',3X,'NOD7',3X,'NOD8',//)
 30 FORMAT (7X, 9(3X,I4))
    RETURN
    END
```

```
      SUBROUTINE OPCOMP (IBP,JA,JB,MAME,NAME,NELCON,NSTART)
C
C     SUBROUTINE OPCOMP CALCULATES THE NAME ARRAY,
C     THE JA ARRAY, AND THE JB ARRAY WHICH
C     ARE USED IN THE OPTIMUM COMPACTING SCHEME
C
C    IBP -- AN ARRAY USED TO STORE THE FUEL-CLAD INTERFACE NODES
C    JA -- AN ARRAY WHICH INDICATES THE NUMBER OF NODES CONTRIBUTING
C         TO THE ITH NODE
C    JB -- THE POINTER ARRAY WHICH INDICATES WHERE THE ITH
C          EQUATION BEGINS IN NAME
C     MAME -- TWO-DIMENSIONAL ARRAY USED TO DEVELOP NAME
C    NAME -- THE ARRAY USED FOR THE OPTIMUM COMPACTING SCHEME
C     NELCON -- CONNECTIVE MATRIX
C     NSTART -- AN ARRAY USED TO STORE CENTERLINE NODAL POINTS
C
      IMPLICIT INTEGER*2(I-N)
      DIMENSION DR(13)
      DIMENSION IBP(1)
      DIMENSION JA(1), JB(1), NAME(1), MAME(26,1)
      DIMENSION NELCON(11,1)
      DIMENSION NSTART(1)
      COMMON/BOUND/ NCOUNT
      COMMON/CONN/ NP,NEL,JC
      COMMON/INP/DR,HEIGHT,NPR,NEZ,NFE
      NELDOF = 8
      WRITE (6,10) NP,NEL,NELDOF
   10 FORMAT(/2X,'NUMNP=',I5,5X,'NUMEL=',I5,5X,'NELDOF=',I5)
      DO 40 I = 1,NP
      JA(I)=1
   40 CONTINUE
      DO 60 I = 1,NP
      DO 50J = 1,26
      MAME(J,I)=0
   50 CONTINUE
   60 CONTINUE
      DO 65 I=1,NP
      MAME(1,I)=I
   65 CONTINUE
C
      DO 100 I = 1,NEL
      DO 90 J=1,NELDOF
      JJ = NELCON(J,I)
      DO 80 K=1,NELDOF
      IF(K.EQ.J) GO TO 80
      KK = NELCON(K,I)
      JA(JJ)=JA(JJ)+1
      JAA=JA(JJ)
```

```
      DO 70 L=2,JAA
      JJL=MAME(L,JJ)
      IF(JJL.EQ.KK) JA(JJ)=JA(JJ)-1
      IF(JJL.EQ.KK) GO TO 80
      IF(JJL.EQ.0) MAME(JAA,JJ)=KK
   70 CONTINUE
   80 CONTINUE
   90 CONTINUE
  100 CONTINUE
C
C *** TO  ACCOUNT FOR THE INTERFACE CONDITIONS
C ***
      NCOUNT = 3
      IBP(1) = NF - 6
      IBP(2) = NFE + 1
      NN = NFE + 7
      NUM = 2*NFE
      WRITE (6,1000) (NSTART(I),I=1,NEZ)
 1000 FORMAT (10X,11I5)
      DO 300 I = 1,NEZ
      IBP(NCOUNT) = NSTART(I) + NUM
      IF (I .EQ. NEZ) GO TO 300
      IBP(NCOUNT + 1) = IBP(NCOUNT) + NN
      NCOUNT = NCOUNT + 2
  300 CONTINUE
      WRITE (6,1000) (IBP(I),I=1,NCOUNT)
      KK = 1
      DO 400 I = 1,NCOUNT
      JJ = IBP(I)
      IF (KK .EQ. 0) GO TO 350
      IF (I .EQ. NCOUNT) GO TO 350
      IF (I .EQ. 1) GO TO 350
      DO 325 J = 1,13
      MAME (J+13,JJ) = MAME (J,JJ+1)
      MAME (J+13,JJ+1) = MAME (J,JJ)
      MAME (J+13,JJ+3) = MAME (J,JJ+4)
      MAME (J+13,JJ+4) = MAME (J,JJ+3)
  325 CONTINUE
      JA(JJ) = JA(JJ) + 13
      JA (JJ+1) = JA(JJ+1) + 13
      JA (JJ+3) = JA(JJ+3) + 13
      JA (JJ+4) = JA(JJ+4) + 13
      KK = 0
      GO TO 400
  350 KK = 1
      DO 375 J = 1,8
      MAME (J+8,JJ ) = MAME (J,JJ+1)
      MAME (J+8,JJ+1 ) = MAME (J,JJ)
```

```
      IF ((I .EQ.1) .OR. (I .EQ. NEZ1)) JJ = JJ + 1
      MAME (J+8,JJ+2) = MAME (J,JJ+3)
      MAME (J+8,JJ+3) = MAME (J,JJ+2)
      IF ((I .EQ.1) .OR. (I .EQ. NEZ1)) JJ = JJ - 1
  375 CONTINUE
      JA (JJ) = JA (JJ) + 8
      JA (JJ+1) = JA (JJ+1) + 8
      IF ((I .EQ.1) .OR. (I .EQ. NEZ1)) JJ = JJ + 1
      JA (JJ+2) = JA (JJ+2) + 8
      JA (JJ+3) = JA (JJ+3) + 8
      IF (I .EQ.1) KK = 0
  400 CONTINUE
C ***
C ***
      JB(1)=1
      JC=0
      DO200 I = 1,NP
      JN=JA(I)
      JB(I+1)=JB(I)+JA(I)
      JC=JC+JA(I)
  200 CONTINUE
C     WRITE(7,5) (JA(I),I=1,NP)
C     WRITE(7,5) (JB(I),I=1,NP),JC
    5 FORMAT(16I5)
  205 FORMAT (29I4)
      WRITE(6,215) JC
  215 FORMAT(/2X,'JC=',I10)
      DO250 I = 1,NP
      JAA=JA(I)
      JBL=JB(I)
      DO 240 J=1,JAA
      JJ=JBL+J-1
      NAME(JJ)=MAME(J,I)
  240 CONTINUE
  250 CONTINUE
C     WRITE(7,5) (NAME(I),I=1,JC)
      RETURN
      END
```

```
      SUBROUTINE INIT (IBP,NELCON,R,Z,Y)
C
C**   INITIAL SETS THE STARTING VALUES
C**   FOR THE TEMPERATURE AND THE NEUTRON FLUX
C
      IMPLICIT INTEGER*2(I-N)
      DIMENSION DR(13)
      DIMENSION IBP(1)
      DIMENSION NELCON(11,1)
      DIMENSION R(1), Z(1)
      DIMENSION Y(7,1)
      COMMON/BOUND/ NCOUNT
      COMMON/CONN/ NP,NEL,JC
      COMMON/INP/DR,HEIGHT,NPR,NEZ,NFE
      K = 0
      DO 100 N = 1,NCOUNT
      IB = IBP(N)
      IF(K .EQ. 1) GO TO 50
      K = 1
      IBF = IB - 2*NFE + NP
      IBCC = IB + 4 + NP
      GO TO 75
   50 IBF = IB - NFE + NP
      IBCC =  IB + 3 + NP
      K = 0
   75 IBC = IB + 1 + NP
C**
C**   INPUT INITIAL TEMPERATURES FOR THE FUEL CENTERLINE, FUEL
C**   SURFACE, CLADDING, AND THE COOLANT
C**   ALONG THE AXIAL DIRECTION
C**
      READ (5,1000) Y(1,IBF), Y(1,IB+NP), Y(1,IBC), Y(1,IBCC)
 1000 FORMAT (4F10.5)
  100 CONTINUE
C**
C**   CALCULATE THE RADIAL TEMPERATURES USING INPUT
C**   TEMPERATURES.  CLAD TEMPERATURES AND THE COOLANT
C**   TEMPERATURES ARE INITIALLY ASSUMED CONSTANT
C**   IN THE RADIAL  DIRECTION.
C**
      DO 600 M = 1,NEL
      IT = NELCON(11,M) + 1
      GO TO (200,300,400,400),IT
  200 DO 250 K = 1,8
      NODE = NELCON(K,M)
      IF (K .GT. 3) GO TO 230
      NODES = NELCON(1,M+1)
      NODEM = NELCON(3,M)
```

```
      A2 = R(NODES)**2
      R2 = R(NODE)**2
      Y(1,NODE+NP) = Y(1,NODEM+NP)*(1.0-R2/A2) + Y(1,NODES+NP)*R2/A2
      GO TO 250
  230 IF((K .EQ. 4) .OR. (K .EQ. 8)) GO TO 240
      NODES =   NELCON(7,M+1)
      NODEM = NELCON(5,M)
      A2 = R(NODES)**2
      R2 = R(NODE)**2
      Y(1,NODE+NP) = Y(1,NODEM+NP)*(1.0-R2/A2) + Y(1,NODES+NP)*R2/A2
      GO TO 250
  240 NODES = NELCON(8,M+1)
      NODEM = NELCON (4,M)
      A2 = R(NODES)**2
      R2 = R(NODE)**2
      Y(1,NODE+NP) = Y(1,NODEM+NP)*(1.0-R2/A2) + Y(1,NODES+NP)*R2/A2
  250 CONTINUE
      GO TO 600
  300 KK = 1
      IF(NFE .EQ. 1) KK = 0
      NODE = NELCON(2,M)
      NODES = NELCON(1,M)
      NODEM = NELCON(3,M-KK)
      A2 = R(NODES)**2
      R2 = R(NODE)**2
      Y(1,NODE+NP) = Y(1,NODEM+NP)*(1.0-R2/A2) + Y(1,NODES+NP)*R2/A2
      NODE = NELCON(6,M)
      NODES = NELCON(7,M)
      NODEM = NELCON(5,M-KK)
      Y(1,NODE+NP) = Y(1,NODEM+NP)*(1.0-R2/A2) + Y(1,NODES+NP)*R2/A2
      GO TO 600
  400 NODE1 = NELCON(1,M) + NP
      NODE2 = NELCON(2,M) + NP
      NODE3 = NELCON(3,M) + NP
      NODE4 = NELCON(4,M) + NP
      NODE5 = NELCON(5,M) + NP
      NODE6 = NELCON(6,M) + NP
      NODE7 = NELCON(7,M) + NP
      NODE8 = NELCON(8,M) + NP
      Y(1,NODE1) = Y(1,NODE3)
      Y(1,NODE2) = Y(1,NODE3)
      Y(1,NODE6) = Y(1,NODE5)
      Y(1,NODE7) = Y(1,NODE5)
      Y(1,NODE8) = Y(1,NODE4)
  600 CONTINUE
C**
C**   INPUT THE MAXIMUM FLUX (I.E. THE FLUX
C**   AT THE AXIAL CENTER)  RADIAL FLUX WILL BE ASSUMED
```

```
C**    TC REMAIN CONSTANT (FLAT FLUX ASSUMPTION)
C**    AXIALLY THE FLUX ASSUMED TC VARY AS THE
C**    SHAPE OF THE SINE FUNCTION
       READ(5,1001) FMAX
 1001  FCRMAT (F10.5)
       DC 700 M = 1,NP
       X = Z(M)*3.1415926/HEIGHT
       S = SIN(X)
       Y(1,M)  = FMAX*S
  700  CCNTINUE
       WRITE (6,1050)
 1050  FCRMAT ('1',9X,'NODAL',5X,'INITIAL',3X,'INITIAL',/,10X,'POINT',5X,
      1  'FLUX',6X,'TEMP',//)
       DC 800 I = 1,NP
       WRITE (6,1100) I, Y(1,I),Y(1,I+NP)
  800  CCNTINUE
 1100  FCRMAT (9X,I4,E13.3,F10.3)
       RETURN
       END
```

```
      SUBROUTINE SHAPE
C
C     SHAPE EVALUATES THE SHAPE FUNCTION AND
C     ITS DERIVATIVE WITH RESPECT TO XI AND
C     WITH RESPECT TO ETA AT EACH OF THE GAUSS POINTS
C     SUBROUTINE SHAPE ALSO SETS THE WEIGHTS AND LOCATION
C     OF THE GAUSS POINTS DEPENDING UPON THE ORDER
C     OF GAUSSIAN QUADRATURE USED
C
      IMPLICIT INTEGER*2(I-N)
      REAL*8 GP,ETA,SN,DESN,DXSN,XI,ETA1,ETA2,XI1,XI2,ETASQ,XISQ
      REAL*8 XW, WT
      DIMENSION SN(8,25), DESN(8,25), DXSN(8,25), WT(25), GP(5), XW(5)
      COMMON/GP/ GP,NORD
      COMMON/SHAFUN/SN, DESN, DXSN
      COMMON/WEIGHT/ WT
      NO = NORD-2
      N = 0
      GO TO (4,6,8),NO
    4 GP(1) = 0.774596669241483
      GP(2) = 0.0
      GP(3) = -GP(1)
      XW(1) = 5.0/9.0
      XW(2) = 8.0/9.0
      XW(3) = XW(1)
      GO TO 10
    6 GP(1) = 0.861136311594053
      GP(2) = 0.339981043584856
      GP(3) = -GP(2)
      GP(4) = -GP(1)
      XW(1) = 0.347854845137454
      XW(2) = 0.652145154862546
      XW(3) = XW(2)
      XW(4) = XW(1)
      GO TO 10
    8 GP(1) = 0.906179845938664
      GP(2) = 0.538469310105683
      GP(3) = 0.0
      GP(4) = -GP(2)
      GP(5) = -GP(1)
      XW(1) = 0.236926885056189
      XW(2) = 0.478628670499366
      XW(3) = 0.568888888888889
      XW(4) = XW(2)
      XW(5) = XW(1)
   10 DO 13 I = 1,NORD
      DO 13 J = 1,NORD
      N = N + 1
```

```
13 WT(N) = XW(I)*XW(J)
   N = 0
   DO 15 I = 1,NORD
   ETA = GP(I)
   DO 15 J = 1,NORD
   XI = GP(J)
   N = N + 1
   ETA1 = 1.0 + ETA
   ETA2 = 1.0 - ETA
   XI1 = 1.0 + XI
   XI2 = 1.0 - XI
   ETASQ = ETA*ETA
   XISQ = XI*XI
   SN(1,N) = 0.25*ETA1*XI1*(XI+ETA-1.0)
   DESN(1,N) = 0.25*XI1*(2.0*ETA+XI)
   DXSN(1,N) = 0.25*ETA1*(2.0*XI+ETA)
   SN(2,N) = 0.5*(1.0-XISQ)*ETA1
   DESN(2,N) = SN(2,N)/ETA1
   DXSN(2,N) = -XI*ETA1
   SN(3,N) = 0.25*XI2*ETA1*(-XI+ETA-1.0)
   DESN(3,N) = 0.25*XI2*(2.0*ETA-XI)
   DXSN(3,N) = 0.25*ETA1*(2.0*XI-ETA)
   SN(4,N) = 0.5*XI2*(1.0-ETASQ)
   DESN(4,N) = -ETA*XI2
   DXSN(4,N) = -SN(4,N)/XI2
   SN(5,N)= -0.25*XI2*ETA2*(XI+ETA+1.0)
   DESN(5,N) = 0.25*XI2*(2.0*ETA+XI)
   DXSN(5,N) = 0.25*ETA2*(2.0*XI+ETA)
   SN(6,N) = 0.5*(1.0-XISQ)*ETA2
   DESN(6,N) = -SN(6,N)/ETA2
   DXSN(6,N) = -XI*ETA2
   SN(7,N) = 0.25*XI1*ETA2*(XI-ETA-1.0)
   DESN(7,N) = 0.25*XI1*(2.0*ETA-XI)
   DXSN(7,N) = 0.25*ETA2*(2.0*XI-ETA)
   SN(8,N) = 0.5*XI1*(1.0-ETASQ)
   DESN(8,N) = -ETA*XI1
15 DXSN(8,N) = SN(8,N)/XI1
   RETURN
   END
```

```
      SUBROUTINE MATK1 (NELCON)
C
C     SUBROUTINE MATK1 CALCULATES THE8X8 ELEMENT K1
C     AND K2 MATRICES FOR THE ELEMENTSBORDERING
C     ONE OF THE TWO INTERFACES
C
      IMPLICIT INTEGER*2(I-N)
      REAL*8 WT,GP,ETA,XSN,X,Z,XW,Y
      DIMENSION GP(5), XSN(5,5), WT(25)
      DIMENSION NELCON(11,1)
      DIMENSION XK1(8,8), XK2(8,8)
      COMMON/CONN/ NP,NEL,JC
      COMMON/GP/ GP,NORD
      COMMON/K1MAT/XK1,XK2
      COMMON /WEIGHT/ WT
      XW = WT(1)**0.5
      DO 15 I = 1,8
      DO 15 J = 1,8
      XK1(I,J) = 0.0
      XK2(I,J) = 0.0
   15 CONTINUE
      DO 30 K = 1,NORD
      ETA = GP(K)
      XSN(1,K) = 0.5*(ETA+1.0)*ETA
      XSN(2,K) =  0.5*(ETA-1.0)*ETA
      XSN(3,K) = 1.0 - ETA*ETA
   30 CONTINUE
      DO 100 I = 1,3
      DO 80 J = I,3
      Z = 0.0
      DO 60 K = 1,NORD
      X = XSN(I,K)
      Y = XSN(J,K)
      Z = Z + X*Y*WT(K)/XW
   60 CONTINUE
      XK1(I,J) = Z
   80 CONTINUE
  100 CONTINUE
      XK1 (1,7) = XK1 (1,2)
      XK1 (1,8) = XK1 (1,3)
      XK1 (7,7) = XK1 (2,2)
      XK1 (7,8) = XK1 (2,3)
      XK1 (8,8) = XK1 (3,3)
  120 XK2(3,3) = XK1(1,1)
      XK2(3,4) = XK1(1,3)
      XK2(3,5) = XK1(1,2)
      XK2(4,4) = XK1(3,3)
      XK2(4,5) = XK1(2,3)
```

```
      XK2(5,5) = XK1(2,2)
  140 DO 160 I = 1,3
      DO 160 J = 2,3
      XK1(I,J) = 0.0
  160 CONTINUE
      DO 180 I = 1,8
      DO 180 J = I,8
      XK1(J,I) = XK1(I,J)
      XK2(J,I) = XK2(I,J)
  180 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE JACOB  (M,NELCON,R,Z)
C
C
C     JACOB EVALUATES THE JACOBIAN MATRIX,
C     ITS INVERSE, AND ITS DETERMINATE
C     FOR EACH OF THE GAUSS POINTS
C
C     M IS NUMBER OF THE ELEMENT UNDER CONSIDERATION
C
      IMPLICIT INTEGER*2(I-N)
      REAL*8 DR,DZ,DETJ,XJ11,XJ22,STJ11,STJ22
      REAL*8 GP
      DIMENSION GP(5)
      DIMENSION NELCON(11,1)
      DIMENSION R(1), Z(1)
      DIMENSION STJ11(25), STJ22(25)
      COMMON/GP/ GP,NORD
      COMMON/JACINV/ STJ11,STJ22,DETJ
      COMMON/CONN/ NP,NEL,JC
C     CALCULATE THE DETERMINATE OF THE JACOBIAN
      L = NELCON(1,M)
      K = NELCON(3,M)
      DR = R(L) -R(K)
      L = NELCON(5,M)
      DZ = Z(K) - Z(L)
      DETJ = 0.25*DR*DZ
      XJ11 = 2.0/DR
      XJ22 = 2.0/DZ
      NO = NORD*NORD
      DO 100 I = 1,NO
C     CALCULATE THE INVERSE OF THE JACOBIAN MATRIX
      STJ11(I) = XJ11
  100 STJ22(I) = XJ22
      RETURN
      END
```

```
      SUBROUTINE MATH12  (M,NELCON,R,Z)
C
C     SUBROUTINE MATH12 CALCULATES THE 8X8 ELEMENTT H12,
C     H3 AND H5 MATRICES AND CALCULATES THE 8X1 VECTOR F
C     H5 IS STORED IN THE H4 ARRAY TO SAVE CORE STORAGE
C
      IMPLICIT INTEGER*2(I-N)
      REAL*8 SN,DESN,DXSN,STJ11,STJ22,DETJ,WT,ZX,ZY,ZZ,ZZZ,X,Y,XX,YY,RR
      REAL*8 XXX,YYY,FF,RD,GP
      DIMENSION GP(5)
      DIMENSION H12(8,8),H3(8,8), F(8)
      DIMENSION H4(8,8)
      DIMENSION NELCON(11,1)
      DIMENSION R(1), Z(1)
      DIMENSION SN(8,25), DESN(8,25), DXSN(8,25), WT(25)
      DIMENSION STJ11(25), STJ22(25)
      COMMON/CONN/ NP,NEL,JC
      COMMON/GP/ GP,NORD
      COMMON/JACINV/ STJ11,STJ22,DETJ
      COMMON /MATRIX/ H12, H3, F
      COMMON/MTRXH4/ H4
      COMMON/SHAFUN/SN, DESN, DXSN
      COMMON /WEIGHT/ WT
      NO = NORD*NORD
      LE = NELCON(11,M)
      DO 100 I = 1,8
      FF = 0.0
      DO 80 J = 1,8
      ZX= 0.0
      ZY = 0.0
      ZZ = 0.0
      ZZZ = 0.0
      DO 60 K = 1,NO
      X = DXSN(I,K)*STJ11(K)
      Y = DXSN(J,K)*STJ11(K)
      XX= DESN(I,K)*STJ22(K)
      YY= DESN(J,K)*STJ22(K)
      XXX = SN(I,K)
      YYY = SN(J,K)
      RR = 0.0
      DO 40 L = 1,8
      LL = NELCON(L,M)
      RD = R(LL)
      RR = RR + SN(L,K)*RD
   40 CONTINUE
      ZX= ZX+ X*Y*RR*WT(K)
      ZY = ZY + XXX*YY*RR*WT(K)
      ZZ = ZZ + XX*YY*RR*WT(K)
```

```
      ZZZ = ZZZ + XXX*YYY*RR*WT(K)
      IF(J .NE. I) GO TO 60
      IF((LE .EQ. 2) .OR. (LE .EQ. 3)) GO TO 60
      FF = FF + XXX*RR*WT(K)
   60 CONTINUE
      H12(I,J) = (ZX + ZZ)*DETJ
      H3(I,J) = ZZZ*DETJ
      H4(I,J) = ZY*DETJ
      F(I) = FF*DETJ
   80 CONTINUE
  100 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE MATH4 (M,NELCON,R,Z,YIT,Y)
C
C     SUBROUTINE MATH4 CALCULATES THE 8X8, TEMPERATURE
C     DEPENDENT ELEMENT H4 MATRIX
C
      IMPLICIT INTEGER*2(I-N)
      REAL*8 SN,DESN,DXSN,WT,XXX,YYY,RR,HH,TEMP,TAUO
      REAL*8XX,YY,TT,RD,GP
      DIMENSION H4(8,8)
      DIMENSION NELCON(11,1)
      DIMENSION R(1),Z(1)
      DIMENSION SN(8,25), DESN(8,25), DXSN(8,25), WT(25), GP(5)
      DIMENSION Y(7,1)
      DIMENSION YIT(1)
      COMMON/CONN/ NP,NEL,JC
      COMMON/GP/ GP,NORD
      COMMON/MTRXH4/ H4
      COMMON/SHAFUN/SN, DESN, DXSN
      COMMON /WEIGHT/ WT
      N = NORD*NORD
      I = NELCON(1,M)
      J = NELCON(3,M)
      K = NELCON(5,M)
      DETJ = 0.25*(R(I)-R(J))*(Z(J)-Z(K))
      DO 100 I = 1,8
      DO 80 J = 1,8
      HH = 0.0
      DO 60 K = 1,N
      XXX = SN(I,K)
      YYY = SN(J,K)
      RR = 0.0
      TEMP = 0.0
      TAUO = 0.0
      DO 40 L = 1,8
      LL = NELCON(L,M)
      RD = R(LL)
      XX= Y(1,LL+NP)
      YY = YIT(LL)
      RR = RR + SN(L,K)*RD
      TEMP = TEMP + SN(L,K)*XX
      TAUO = TAUO + SN(L,K)*YY
   40 CONTINUE
      IF (TEMP .LE. 1.) TEMP = 1.0
      IF(TAUO .LE. 1.)TAUO=1.0
      TAU1 = TEMP/TAUO
      IF (TAU1 .GE. 0.) GO TO 50
      WRITE(6,999) I,J,M,TEMP,TAUO,RR
  999 FORMAT(5X,3I10,5X,3G18.4)
```

```
      DO 45 II = 1,8
      LL = NELCON(II,M)
      WRITE(6,888) SN(II,K),Y(1,LL+NP),YIT(LL)
  888 FORMAT(5X,3G18.4,//)
   45 CONTINUE
   50 TAUNL = ALOG(TAU1)
      TT = TAUNL
      HH = HH + XXX*YYY*RR*WT(K)*TT
   60 CONTINUE
      H4(I,J) = HH*DETJ
   80 CONTINUE
  100 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SYSH12 (M,BIGH12,BIGH3,BIGH5,BIGF,JA,JB,NAME,NELCON)
C
C     SUBROUTINE SYSH12 DEVELOPES THE SYSTEM MATRICES
C     BIGH12, BIGH3, BIGH5, AND BIGF  FROM THE ELEMENT
C     H12, H3, H4,AND F MATRICES ,RESPECTIVELY
C
C     BIGH12 -- SYSTEM MATRIX FOR THE LAPLACIAN TERM
C               D NI/DR X D NJ/DR + D NI/DZ X D NJ/DZ
C     BIGH3 -- SYSTEM MATRIX FOR THE NI X NJ TERM
C     BIGH5 -- SYSTEM MATRIX FOR THE NI X D NJ/DZ TERM
C     BIGF -- SYSTEM VECTOR FOR THE INITIAL DELAYED NEUTRON EFFECT
      IMPLICIT INTEGER*2(I-N)
      DIMENSION BIGH12(1), BIGH3(1), BIGH5(1)
      DIMENSION BIGF(1)
      DIMENSION H12(8,8),H3(8,8), F(8)
      DIMENSION H4(8,8)
      DIMENSION JA(1), JB(1), NAME(1)
      DIMENSION NELCON(11,1)
      COMMON/CONN/ NP,NEL,JC
      COMMON /MATRIX/ H12, H3, F
      COMMON/MTRXH4/ H4
      DO 120 K = 1,8
      KK = NELCON(K,M)
      KKK = JA(KK)
      LLL = JB(KK) - 1
      DO 100 I = 1,8
      II = NELCON(I,M)
      DO 60 L = 1,KKK
      LL = LLL + L
      KKM = NAME(LL)
      IF (II .EQ. KKM) GO TO 80
   60 CONTINUE
   80 BIGH12(LL) = BIGH12(LL) + H12(K,I)
      BIGH3(LL) = BIGH3(LL) + H3(K,I)
      BIGH5(LL) = BIGH5(LL) + H4(K,I)
  100 CONTINUE
      BIGF(KK) = BIGF(KK) + F(K)
  120 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SYSH4 (M,JA,JB,NAME,NELCON,BIGH4)
C     SYSH4 DEVELOPES THE SYSTEM H4 MATRIX
C     FROM THE  ELEMENT H4 MATRICES
C
C     BIGH4 -- SYSTEM MATRIX FOR  THE TEMPERATURE DEPENDENT MATRIX
C              NATURAL LOG(T/TO) X NI X NJ
      IMPLICIT INTEGER*2(I-N)
      DIMENSION BIGH4(1)
      DIMENSION H4(8,8)
      DIMENSION JA(1), JB(1), NAME(1)
      DIMENSION NELCON(11,1)
      COMMON/CONN/ NP,NEL,JC
      COMMON/MTRXH4/ H4
   50 DO 120 K = 1,8
      KK = NELCON(K,M)
      KKK = JA(KK)
      LLL = JB(KK) - 1
      DO 100 I = 1,8
      II = NELCON(I,M)
      DO 60 L = 1,KKK
      LL = LLL + L
      KKM = NAME(LL)
      IF (II .EQ. KKM) GO TO 80
   60 CONTINUE
   80 BIGH4(LL) = BIGH4(LL) + H4(K,I)
  100 CONTINUE
  120 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SYSK12  (M,JA,JB,NAME,NELCON,R,Z,BIGK)
C
C     SYSK DEVELOPES THE SYSTEM K MATRIX
C     FROM THE ELEMENT MATRICES
C
C     BIGK -- SYSTEM MATRIX FOR THE THERMAL INTERFACE CONDITIONS
C
      IMPLICIT INTEGER*2(I-N)
      DIMENSION BIGK(1)
      DIMENSION JA(1), JB(1), NAME(1)
      DIMENSION NDUM(3)
      DIMENSION NELCON(11,1)
      DIMENSION R(1), Z(1)
      DIMENSION XK1(8,8), XK2(8,8)
      COMMON/CONN/ NP,NEL,JC
      COMMON/K1MAT/XK1,XK2
      NDUM(1) = 1
      NDUM(2) = 7
      NDUM(3) = 8
      MM = NELCON(11,M)
      GO TO (1,1,2),MM
    1 DO 120 K = 1,3
      N = NDUM(K)
      KK = NELCON(N,M)
      KKK = JA(KK)/2
      LLL = JB(KK) - 1
      DO 100 I = 1,3
      NN = NDUM(I)
      II = NELCON(NN,M)
      JJ = II + 1
      N1 = 0
      N2 = 0
      DO 80 L = 1,KKK
      LL = LLL+ L
      LS = LL + KKK
      KKM = NAME(LL)
      KKS = NAME(LS)
      IF(II .NE. KKM) GO TO 60
      BIGK(LL) = BIGK(LL) + XK1(N,NN)
      N1 = 1
   60 IF(JJ .NE. KKS) GO TO 70
      BIGK(LS) = BIGK(LS) - XK1(N,NN)
      N2 = 1
   70 IF((N1 .EQ. 1) .AND. (N2 .EQ. 1)) GO TO 100
   80 CONTINUE
  100 CONTINUE
  120 CONTINUE
      IF (MM .EQ. 1) GO TO 4
```

```
    2 JE = NELCCN(3,M)
      CC 220 K = 3,5
      KK = NELCCN(K,M)
      KKK = JA(KK)/2
      LLL = JB(KK) - 1
      CC 200 I = 3,5
      II = NELCCN(I,M)
      JJ = II + 1
      N1 = 0
      N2 = 0
      CC 180 L = 1,KKK
      LL = LLL + L
      LS = LL + KKK
      KKM = NAME(LL)
      KKS = NAME(LS)
      IF(II .NE. KKM) GC TO 160
      BIGK(LL) = BIGK(LL) + XK2(K,I)
      N1 = 1
  16C IF(JJ .NE. KKS) GC TO 170
      BIGK(LS) = BIGK(LS) - XK2(K,I)
      N2 = 1
  17C IF((N1 .EQ. 1) .ANC. (N2 .EC. 1)) GO TO 2C0
  18C CCNTINUE
  20C CCNTINUE
  22C CCNTINUE
    4 RETURN
      ENC
```

```
      SUBROUTINE DIFFUN(Y,YL,T,HINV,DY,BIGH12,BIGH3,BIGH4,BIGH5,BIGK,
     1 BIGF,IBP,ITYPE,JA,JB,NAME,NELCON,R,Z,YIT)
C
C     SUBROUTINE DIFFUN FORMS THE NODAL EQUATIONS.
C     IT SETS THE ITH EQUATION FOR FLUX EQUAL TO DYI
C     AND FOR TAMPERATURE TO DYII
C
      IMPLICIT INTEGER*2(I-N)
      REAL*8 DYI,DYII
      DIMENSION BIGH12(1), BIGH3(1), BIGH5(1)
      DIMENSION BIGH4(1)
      DIMENSION BIGK(1)
      DIMENSION BIGF(1)
      DIMENSION YIT(1)
      DIMENSION DR(13)
      DIMENSION IBP(1), ITYPE(1)
      DIMENSION JA(1), JB(1), NAME(1)
      DIMENSION NELCON(11,1)
      DIMENSION R(1),Z(1)
      DIMENSION Y(7,1), DY(1)
      COMMON/BOUND/ NCOUNT
      COMMON/CLAD/ DCC,CPC,DENC,TKC,SIGAC,HGAP
      COMMON/CONN/ NP,NEL,JC
      COMMON/COOL/DCCO,SIGACC,VCO,HSURF
      COMMON/FUEL/DCF,B,E,BETA,BETAI,DCLAMI,FFLUXC,AKINF,VEL,
     1   SIGAF1,SIGFF1,SIGAB,SIGFB,DENF
      COMMON/INP/DR,HEIGHT,NPR,NEZ,NFE
      COMMON/TIME/ F,G,RHO
      DATA TOLD/-1.0/
C***  CALCULATE THE TIME DEPENDENT TERMS
      IF (T .EQ. TOLD) GO TO 75
      CALL FUNC (T,HINV)
C***  CALCULATE THE SYSTEM H4 MATRIX
C***  WHICH IS TEMPERATURE DEPENDENT
      DO 30 I = 1,JC
   30 BIGH4(I) = 0.0
      DO 50 I = 1,NEL
      IF (NELCON(11,I) .GT. 1) GO TO 50
      CALL MATH4 (I,NELCON,R,Z,YIT,Y)
      CALL SYSH4 (I,JA,JB,NAME,NELCON,BIGH4)
   50 CONTINUE
      TOLD = T
   75 VINV = 1.0/VEL
      Z1=Z(IBP(2))-Z(IBP(1))
      B1 = 1.0 - BETA
      SIGFF = SIGFF1
      SIGAF = SIGAF1
      CONST1 = AKINF*B1 - 1.0
```

```
      CONST2 = SIGAF*B1*RHO
      TD = 0.98
      HMCD = 1.0/(0.55*HEIGHT)
      DO 510 I = 1,NP
      JAA = JA(I)
      JBB = JB(I)
      JAB = JAA + JBB - 1
      IT = ITYPE(I) + 1
      II = NP + I
      DYI = 0.0
      DYII = 0.0
      GO TO (100,200,300,400),IT
  100 DO 150 J = JBB,JAB
      NN = NAME(J)
      NNN = NN + NP
      DYI = DYI + (DCF*DBLE(BIGH12(J))-((CONST1+F)*SIGAF+CONST2)*
     1DBLE(BIGH3(J))+B*SIGAF*B1*DBLE(BIGH4(J)))*Y(1,NN)-G*DBLE(BIGF(NN))
     2 + VINV*HINV*DBLE(BIGH3(J))*Y(2,NN)
      TDK = Y(1,NNN) + 273.
      TKF=((1.0-2.5*(1.0-TD))*(45.1/(135.+TDK) + 4.79E-13*TDK**3))*0.239
      CPF =(18.45 + 2.431E-3*TDK - 2.272E5/TDK**2)/270.07
      DYII = DYII + TKF*DBLE(BIGH12(J))*Y(1,NNN) + HINV*DENF*CPF*
     1 DBLE(BIGH3(J))*Y(2,NNN)-E*SIGFF*DBLE(BIGH3(J))*Y(1,NN)
  150 CONTINUE
      GO TO 500
C
C     DETERMINE THE TYPE OF NODE BEING CONSIDERED
C     K = 0     NODE NOT ON THE FUEL-CLAD INTERFACE
C     K = 1     NODE ON THE FUEL-CLAD INTERFACE
C
  200 K = 0
      DO 210 IB = 1,NCOUNT
      IF(IBP(IB) .EQ. I) GO TO 215
  210 CONTINUE
      GO TO 220
  215 K = 1
  220 DO 250 J = JBB,JAB
      NN = NAME(J)
      NNN = NN + NP
      DYI = DYI + (DCF*DBLE(BIGH12(J))-((CONST1+F)*SIGAF+CONST2)*
     1DBLE(BIGH3(J))+B*SIGAF*B1*DBLE(BIGH4(J)))*Y(1,NN)-G*DBLE(BIGF(NN))
     2 + VINV*HINV*DBLE(BIGH3(J))*Y(2,NN)
      RA = R(NN)
      TDK = Y(1,NNN) + 273.
      TKF=((1.0-2.5*(1.0-TD))*(45.1/(135.+TDK) + 4.79E-13*TDK**3))*0.239
      CPF =(18.45 + 2.431E-3*TDK - 2.272E5/TDK**2)/270.07
      HGAP =(1000.0 + 247.0*COS(3.14159*(Z(NN)*HMCD-0.81818)))*1.356E-4
      CONST3 = HGAP*Z1
```

```
240 DYII = DYII + (TKF*DBLE(BIGH12(J))+RA*CONST3*DBLE(BIGK(J)))*
   1 Y(1,NNN) + DENF*CPF*HINV*DBLE(BIGH3(J))*Y(2,NNN)
   2 - E*SIGFF*DBLE(BIGH3(J))*Y(1,NN)
    IF(K .EQ. 0) GO TO 250
    JJ = J + JAA/2
    NM = NAME(JJ)
    DYI = DYI + (DCC*DBLE(BIGH12(JJ)) + SIGAC*DBLE(BIGH3(JJ)))*Y(1,NM)
   1 + VINV*HINV*DBLE(BIGH3(JJ))*Y(2,NM)
250 CONTINUE
    GO TO 500
C
C   DETERMINE THE TYPE OF NODE BEING CONSIDERED
C   K = 1     NODE NOT ON AN INTERFECE
C   K = 2     NODE ON THE FUEL-CLAD INTERFACE
C   K = 3     NODE ON THE CLAD-COOLANT INTERFACE
C
300 K = 1
    NODD = 0
    DO 320 IB = 1,NCOUNT
    LBP = IBP(IB) + 1
    IF(LBP .EQ. I) GO TO 325
    NBP = LBP + 2
    IF(NODD .EQ. 0) GO TO 305
    NBP = NBP - 1
    NODD = 0
    GO TO 310
305 NODD = 1
310 IF(NBP .EQ. I) GO TO 330
320 CONTINUE
    GO TO 340
325 K = 2
    GO TO 340
330 K = 3
340 DO 380 J = JBB,JAB
    NN = NAME(J)
    NNN = NN + NP
    IF(K .EQ. 2) GO TO 360
    DYI = DYI + (DCC*DBLE(BIGH12(J))  + SIGAC*DBLE(BIGH3(J))) *Y(1,NN)
   1 + VINV*HINV*DBLE(BIGH3(J)) *Y(2,NN)
360 HGAP =(1000.0 + 247.0*COS(3.14159*(Z(NN)*HMCD-0.81818)))*1.356E-4
    H = HGAP
    IF(K .EQ. 3)H =  HSURF
    RA = R(NN)
    DYII = DYII + (TKC*DBLE(BIGH12(J)) + RA*H*Z1*DBLE(BIGK(J)))*
   1Y(1,NNN) +  DENC*CPC*DBLE(BIGH3(J))*HINV*Y(2,NNN)
    IF(K .NE. 3) GO TO 380
    JJ = J + JAA/2
    NM = NAME(JJ)
```

```
      DYI = DYI + (DCCO*DBLE(BIGH12(JJ)) + SIGACO*DBLE(BIGH3(JJ)))*
     1Y(1,NM) +  VINV*HINV*DBLE(BIGH3(JJ))*Y(2,NM)
  380 CONTINUE
      IF(K .NE. 2) GO TO 500
      DYI = Y(1,I) - Y(1,I-1)
      GO TO 500
C
C     DETERMINE THE TYPE OF NODE BEING CONSIDERED
C     K = 1    NODE NOT ON CLAD-COOLANT INTERFACE
C     K = 2    NODE ON THE CLAD-COOLANT INTERFACE
  400 K = 1
      NODD = 0
      DO 420 IB = 1,NCOUNT
      LBP = IBP(IB) + 4
      IF(NODD .EQ. 0) GO TO 410
      LBP = LBP - 1
      NODD = 0
      GO TO 415
  410 NODD = 1
  415 IF(LBP .EQ. I) GO TO 425
  420 CONTINUE
      GO TO 430
  425 K = 2
  430 DO 480 J = JBB,JAB
      NN = NAME(J)
      NNN = NN + NP
      IF(K .EQ. 2) GO TO 450
      DYI = DYI + (DCCO*DBLE(BIGH12(J))  + SIGACC*DBLE(BIGH3(J)))*
     1Y(1,NN) +  VINV*HINV*DBLE(BIGH3(J)) *Y(2,NN)
  450 RE = R(NN)
      TDF = 1.8*Y(1,NNN) + 32.0
      TKCO = (54.306 - 1.878E-2*TDF + 2.0914E-6*TDF**2)*4.134E-3
      DENCO = (59.566 - 7.9504E-3*TDF - 0.2872E-6*TDF**2 +
     1   0.06035E-9*TDF**3)*0.01601
      CPCO = 0.34574 -0.79226E-4*TDF + 0.34086E-7*TDF**2
      DYII = DYII + (TKCO*DBLE(BIGH12(J)) + RB*HSURF*Z1*DBLE(BIGK(J)) +
     1 VCO*DENCO*CPCO*DBLE(BIGH5(J)))*Y(1,NNN) + DENCO*CPCO*
     2DBLE(BIGH3(J))*HINV*Y(2,NNN)
  480 CONTINUE
      IF(K .NE. 2) GO TO 500
      DYI = Y(1,I) - Y(1,I-1)
  500 DY(I) = DYI
      DY(II) = DYII
  510 CONTINUE
C**
C**
C     TO ACCOUNT FOR THE BOUNDARY CONDITIONS
C**
```

```
C**
      NFF = NF + 1
      NN = NPR*2
      NN = NPF - NN
      DO 600 I = NN,NP
      DY(I) = 0.0
  600 CONTINUE
C**
C**
C     TO ACCOUNT FOR THE TEMP BOUNDARY CONDITIONS
C     (CONSTANT COOLANT INLET TEMPERATURES)
C**
C**
      NN = NP - 2
      DO 700 I = NN,NP
      DY(I+NP) = 0.0
  700 CONTINUE
      RETURN
      END
```

```
      SLEROUTINE FUNC(T,HINV)
C
C***  CALCULATES TIME DEPENCENT FUNCTICNS
C***  FCR THIS ANALYSIS DELAYED NEUTRCN GRCUPS
C***  TAKEN TC BE ONE AVERAGED GROUP
C
      CCMMCN/FUEL/DCF,B,E,BETA,BETAI,CCLAMI,FFLUXC,AKINF,VEL,
     1   SIGAF,SIGFF,SIGAB,SIGFB,DENF
      CCMMCN/FEACT/ RHOA,RHCB
      CCMMON/TIME/ F,G,RHC
      CATA ECLC/1.0/, TCLD/C.0/, FI/0.C/
C***  CALCULATE REACTIVITY INPUT
      RHC = RHOA + RHOB*T
      RHCLD = RHCA + RHCB*TOLC
C**   CALCULATE THE G FUNCTICN
      C1I = BETAI*AKINF*SIGAF*FFLUXO/CCLAMI
      X = DCLAMI*T
      EX= EXP(X)
      EE = EXP(-X)
      G = C1I*DCLAMI*EE
C***  CALCULATE THE F FUNCTICN
      C = DCLAMI*BETAI*EE
      CC = AKINF + RHO
      CCCLD = AKINF + RHCLD
      FI = FI + 0.5*(EOLD*CCOLD +EX*CC)/HINV
      F = C*FI
      TCLD = T
      ECLC = EX
      RETURN
      ENC
```

```
      SUBROUTINE JACMAT (Y,YL,T,HINV,A2,N,NY,EPS,DY,F1,PW,BIGH12,BIGH3,
     1 BIGH4,BIGH5,BIGK,IBP,ITYPE,JA,JB,NAME,NELCON,R,Z)
C
C     SUBROUTINE JACMAT CALCULATES THE JACOBIAN MATRIX.
C     FLUX JACOBIAN STORED IN THE FIRST JC ELEMENTS OF THE
C     PW ARRAY AND THE TEMPERATURE JACOBIAN STORED IN
C     THE SECOND JC ELEMENTS OF THE PW ARRAY
C
      IMPLICIT INTEGER*2(I-N)
      INTEGER*4 N,NY
      DIMENSION BIGH12(1), BIGH3(1), BIGH5(1)
      DIMENSION BIGH4(1)
      DIMENSION BIGK(1)
      DIMENSION DR(13)
      DIMENSION IBP(1), ITYPE(1)
      DIMENSION JA(1), JB(1), NAME(1)
      DIMENSION NELCON(11,1)
      DIMENSION R(1),Z(1)
      DIMENSION Y(7,1), DY(1), PW(1)
      COMMON/BOUND/ NCOUNT
      COMMON/CLAD/ DCC,CPC,DENC,TKC,SIGAC,HGAP
      COMMON/CONN/ NP,NEL,JC
      COMMON/COOL/DCCO,SIGACC,VCO,HSURF
      COMMON/FUEL/DCF,B,E,BETA,BETAI,DCLAMI,FFLUXC,AKINF,VEL,
     1    SIGAF1,SIGFF1,SIGAB,SIGFB,DENF
      COMMON/INP/DR,HEIGHT,NPR,NEZ,NFE
      COMMON/TIME/ F,G,RHO
      AH = -A2*HINV
      Z1=Z(IBP(2))-Z(IBP(1))
      VINV = 1.0/VEL
      B1 = 1.0 - BETA
      SIGFF = SIGFF1
      SIGAF = SIGAF1
      CONST1 = AKINF*B1 - 1.0
      CONST2 = SIGAF*B1*RHO
      TC = 0.98
      HMOD = 1.0/(0.55*HEIGHT)
      DO 500 I = 1,NP
      JAA = JA(I)
      JBB = JB(I)
      JAB = JAA + JBB - 1
      JA2 = JAA/2
      JA3 = JA2 + JBB
      IT = ITYPE(I) + 1
      GO TO (100,200,300,400),IT
  100 DO 150 J = JBB,JAB
      NNN = NAME(J) + NP
      PW(J) = DCF*BIGH12(J)-((CONST1+F)*SIGAF+CONST2-AH*VINV)*BIGH3(J)
```

```
     1 + B*SIGAF*B1*BIGH4(J)
      TDK = Y(1,NNN) + 273.
      TKF=((1.0-2.5*(1.0-TD))*(45.1/(135.+TDK) + 4.79E-13*TDK**3))*0.239
      CPF =(18.45 + 2.431E-3*TDK - 2.272E5/TDK**2)/270.07
      PW(J+JC) = TKF*BIGH12(J)+DENF*CPF*AH*BIGH3(J)
  150 CONTINUE
      GO TO 500
C
C     DETERMINE THE TYPE OF NODE BEING CONSIDERED
C     K = 0    NODE NOT ON THE FUEL-CLAD INTERFACE
C     K = 1    NODE ON THE FUEL-CLAD INTERFACE
  200 K = 0
      DO 210 IB = 1,NCOUNT
      IF(IBP(IB) .EQ. I) GO TO 215
  210 CONTINUE
      GO TO 220
  215 K = 1
  220 DO 250 J = JBB,JAB
      PW(J) = DCF*BIGH12(J)-((CONST1+F)*SIGAF+CONST2-AH*VINV)*BIGH3(J)
     1 + B*SIGAF*B1*BIGH4(J)
      NN = NAME(J)
      NNN = NAME(J) + NP
      RA = R(NN)
      HGAP =(1000.0 + 247.0*COS(3.14159*(Z(NN)*HMCD-0.81818)))*1.356E-4
      CONST3 = HGAP*Z1
      TDK = Y(1,NNN) + 273.
      TKF=((1.0-2.5*(1.0-TD))*(45.1/(135.+TDK) + 4.79E-13*TDK**3))*0.239
      CPF =(18.45 + 2.431E-3*TDK - 2.272E5/TDK**2)/270.07
      PW(J+JC) = TKF*BIGH12(J)+DENF*CPF*AH*BIGH3(J)+RA*CONST3*BIGK(J)
      IF(K .EQ. 0) GO TO 250
      IF(J .LT. JA3) GO TO 250
      JJ = JAB + J - JA3 + 1
      PW(J) = DCC*BIGH12(JJ)+ (SIGAC + AH*VINV)*BIGH3(JJ)
      IF(J .NE. JA3) GO TO 250
      PW(JBB) = PW(JBB) + PW(J)
      PW(J) = 0.0
  250 CONTINUE
      GO TO 500
C
C     DETERMINE THE TYPE OF NODE BEING CONSIDERED
C     K = 1    NODE NOT ON AN INTERFACE
C     K = 2    NODE ON THE FUEL-CLAD INTERFACE
C     K = 3    NODE ON THE CLAD-COOLANT INTERFACE
C
  300 K = 1
      NCCC = 0
      DO 320 IB = 1,NCOUNT
      LEF = IBP(IB) + 1
```

```
      IF(LBP .EQ. 1) GO TO 325
      NBP = LBP + 2
      IF (NCDD .EQ. 0) GO TO 305
      NBP = NBP - 1
      NCDD = 0
      GO TO 310
  305 NCDD = 1
  310 IF(NBP .EQ. I) GO TO 330
  320 CONTINUE
      GO TO 340
  325 K = 2
      GO TO 340
  330 K = 3
  340 DO 380 J = JBB,JAB
      NN = NAME(J)
      NNN = NAME(J) + NP
      RA = R(NN)
      PW(J) = 0.0
      IF(K .EQ.2) GO TO 360
      PW(J) = DCC*BIGH12(J) + (SIGAC + AH*VINV)*BIGH3(J)
  360 HGAP =(1000.0 + 247.0*COS(3.14159*(Z(NN)*HMCD-0.81818)))*1.356E-4
      H = HGAP
      IF(K .EQ. 3) H =  HSURF
      PW(J+JC) = TKC*BIGH12(J)+RA*H*Z1*BIGK(J)+AH*DENC*CPC*BIGH3(J)
      IF(K .NE. 3) GO TO 380
      IF(J .LT. JA3) GO TO 380
      JJ = JAB + J - JA3 + 1
      PW(J) = DCCO*BIGH12(JJ) + (SIGACC + AH*VINV)*BIGH3(JJ)
      IF(J .NE. JA3) GO TO 380
      PW(JBB) = PW(JBB) + PW(J)
      PW(J) = 0.0
  380 CONTINUE
      IF(K .NE. 2) GO TO 500
      PW(JBB) = 1.0
      PW(JA3) = -1.0
      GO TO 500
C
C     DETERMINE THE TYPE OF NODE BEING CONSIDERED
C     K = 1    NODE NOT ON CLAD-COOLANT INTERFACE
C     K = 2    NODE ON THE CLAD-COOLANT INTERFACE
C
  400 K = 1
      NCDD = 0
      DO 420 IB = 1,NCOLNT
      LBP = IBP(IB) + 4
      IF(NCDD .EQ. 0) GO TO 410
      LBP = LBP - 1
      NCDD = 0
```

```
      GO TO 415
  410 NCCC = 1
  415 IF(LBP .EQ. I) GO TO 425
  420 CONTINUE
      GO TO 430
  425 K = 2
  430 DO 480 J = JBB,JAB
      NN = NAME(J)
      NNN = NAME(J) + NP
      PW(J) = 0.0
      IF(K .EQ. 2) GO TO 450
      PW(J) = DCCO*BIGH12(J) + (SIGACC + AH*VINV)*BIGH3(J)
  450 RB = R(NN)
      TDF = 1.8*Y(1,NNN) + 32.0
      TKCO = (54.306 - 1.878E-2*TDF + 2.0914E-6*TDF**2)*4.134E-3
      CPCO = 0.34574 -0.79226E-4*TDF + 0.34086E-7*TDF**2
      DENCO = (59.566 - 7.9504E-3*TDF - 0.2872E-6*TDF**2 +
     1   0.06035E-9*TDF**3)*0.01601
      A = DENCO*CPCO
      PW(J+JC) = TKCO*BIGH12(J) + VCO*A*BIGH5(J) + AH*A*BIGH3(J)
     1 + RB*HSURF*Z1*BIGK(J)
  480 CONTINUE
      IF(K .NE. 2) GO TO 500
      PW(JBB) = 1.0
      PW(JA3) = -1.0
  500 CONTINUE
C**
C**
C     TO ACCOUNT FOR THE FLUX AXIAL BOUNDARY CONDITIONS
C**
C**
      NPF = NP + 1
      NN = NPF*2
      NN = NPF - NN
      DO 600 I = NN,NP
      PW(JB(I)) = 1.0
      JS = JB(I) + 1
      JE = JB(I) + JA(I) - 1
      DO 600 J = JS,JE
      PW(J) = 0.0
  600 CONTINUE
C**
C**
C     TO ACCOUNT FOR THE TEMP BOUNDARY CONDITIONS
C     (CONSTANT COOLANT INLET TEMPERATURES)
C**
C**
      NN = NP - 2
```

```
      DO 700 I = NN,NP
      PW(JB(I)+JC) = 1.0
      JS = JB(I) + 1 + JC
      JE = JB(I) + JA(I) - 1 + JC
      DO 700 J = JS,JE
      PW(J) = 0.0
  700 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE NUITSL (PW,DY,F1,N,NY,EPS,YMAX,NEWPW,KRET,BIGH3,
     1 ITYPE,JA,JB,NAME,NELCON)
C
C     SUBROUTINE NUITSL CALCULATES THE NEWTON-ITERATES
C     USING A 'SOR' TECHNIQUE WITH AN OVER-RELAXATION
C     FACTOR EQUAL TO OMEGA1
C
      IMPLICIT INTEGER*2(I-N)
      INTEGER*4 N,NY,NEWPW,KRET
      REAL*8 PN,TN
      DIMENSION BIGH3(1)
      DIMENSION ITYPE(1)
      DIMENSION JA(1), JB(1), NAME(1)
      DIMENSION NELCON(11,1)
      DIMENSION PW(1), DY(1), F1(1), YMAX(1)
      COMMON/CONN/ NP,NEL,JC
      COMMON/FUEL/CCF,B,E,BETA,BETAI,CCLAMI,FFLUXC,AKINF,VEL,
     1    SIGAF,SIGFF,SIGAB,SIGFB,DENF
      DATA OMEG,OMEGM1/1.02,0.02/
      TEST = 1.E35
      KRET = 0
      EPSS = EPS**2
      EPSA2 = EPSS*0.0001
      NCIT = N
      E1 = E*SIGFF
      DO 100 I = 1,NP
      F1(I) = DY(I)/PW(JB(I))
  100 F1(I+NP) = DY(I+NP)/PW(JB(I)+JC)
      DO 300 ITT = 1,NCIT
      RCH = 0.
      CH = 0.
      RCH1 = 0.
      RCH2 = 0.
      CH1 = 0.
      CH2 = 0.
      DO 200 I = 1,NP
      JS = JB(I) + 1
      JE = JB(I+1) - 1
      IT = ITYPE(I)
      PN = DY(I)
      TN = DY(I+NP)
      DO 150 J = JS,JE
      PN = PN - DBLE(PW(J))*F1(NAME(J))
      TN = TN - DBLE(PW(J+JC))*F1(NAME(J)+NP)
      IF(IT .GT. 1) GO TO 150
      TN = TN + E1*DBLE(BIGH3(J))*F1(NAME(J))
  150 CONTINUE
      PN = PN/PW(JS-1)
```

```
      TN = TN/PW(JS+JC-1)
      FN = FN*OMEG - F1(I)*OMEGM1
      TN = TN*OMEG - F1(I+NP)*OMEGM1
      PCH = F1(I) - PN
      TCH = F1(I+NP) - TN
      IF((ABS(PCH).GT.TEST).OR.(ABS(TCH).GT.TEST))GO TO 180
      CH1 = CH1 + (PCH/YMAX(I))**2
      CH2 = CH2 + (TCH/YMAX(I+NP))**2
      RCH1 = RCH1 + (PCH/AMAX1(ABS(SNGL(PN)),EPS))**2
      RCH2 = RCH2 + (TCH/AMAX1(ABS(SNGL(TN)),EPS))**2
      GO TO 190
  180 CH1 = 1.
      CH2 = 1.
      RCH1 = 1.
      RCH2 = 1.
  190 F1(I) = PN
  200 F1(I+NP) = TN
      CH = CH1 + CH2
      RCH = RCH1 + RCH2
      IF(RCH .LT. EPSS) RETURN
      IF(CH .LT. EPSA2) RETURN
  300 CONTINUE
      KRET = 1
      RETURN
      END
```

```
      SUBROUTINE LDASUB (Y,YL,T,TEND,N,NY,M,JSTART,KFLAG,MAXOR,IPRT,H,
     1 HMIN,HMAX,RMSEPS,SAVE,YLSV,YMAX,ER,ESV,F1,DY,PW,BIGH12,BIGH3,
     2 BIGH4,BIGH5,BIGK,BIGF,IBP,ITYPE,JA,JB,NAME,NELCON,RR,Z,YIT)
C
C     SUBROUTINE LDASUB IS A MODIFICATION OF SUBROUTINE DFASUB
C     WHICH IS DUE TO R. L. BROWN AND C. W. GEAR.  DFASUB IS DOCUMENTED
C     IN THE REPORT
C        DOCUMENTATION FOR DFASUB--
C        BY R. L. BROWN AND C. W. GEAR
C        REPORT UIUCDCS-R-73-575, JULY 1973
C        UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
C        URBANA, ILLINOIS  61801
C     THIS REPORT IS AVAILABLE FROM THE NATIONAL TECHNICAL INFORMATION
C     SERVICE OF THE U. S. DEPARTMENT OF COMMERCE UNDER ACCESSION NUMBER
C     COO-1469-225.
C
C     THE MODIFICATION HERE IS DOCUMENTED IN THE REPORT
C        A PROGRAM FOR THE NUMERICAL SOLUTION OF LARGE SPARSE SYSTEMS OF
C        ALGEBRAIC AND IMPLICITLY DEFINED STIFF DIFFERENTIAL EQUATIONS
C        BY RICHARD FRANKE
C        REPORT NPS53FE76051, MAY 1976
C        NAVAL POSTGRADUATE SCHOOL
C        MONTEREY, CALIFORNIA  93940
C
C     ------------------------------------------------------------------
C
C     THE CALLING SEQUENCE FOR LDASUB IS
C
C     CALL LDASUB(Y,YL,T,TEND,N,NY,M,JSTART,KFLAG,MAXOR,IPRT,H,HMIN,
C     HMAX,RMSEPS,SAVE,YLSV,YMAX,ER,ESV,F1,DY,PW)
C
C     WHERE THE PARAMETERS ARE DEFINED AS FOLLOWS.
C        Y          - ARRAY DIMENSIONED (7,NY).  THIS ARRAY CONTAINS THE
C                     DEPENDENT VARIABLES AND THEIR SCALED DERIVATIVES.
C                     Y(J+1,I) CONTAINS THE J-TH DERIVATIVE OF THE I-TH VAR
C                     IABLE TIMES H**J/J-FACTORIAL, WHERE H IS THE CURRENT
C                     STEPSIZE.  ON FIRST ENTRY THE CALLER SUPPLIES THE
C                     INITIAL VALUES OF EACH VARIABLE IN Y(1,I) AND AN
C                     ESTIMATE OF THE INITIAL VALUES OF THE DERIVATIVES
C                     IN Y(2,I).  ON SUBSEQUENT ENTRIES IT IS ASSUMED THAT
C                     THE ARRAY HAS NOT BEEN CHANGED.  TO INTERPOLATE TO
C                     NON-MESH POINTS, THESE VALUES CAN BE USED AS FOLLOWS.
C                     IF H IS THE CURRENT STEPSIZE AND VALUES AT TIME T+E
C                     NEEDED, LET S = E/H  AND THEN
C
C                                                NQ
C                  I-TH VARIABLE AT  T+E IS  SUM Y(J+1,I)*S**J
C                                                J=0
```

```
C                                                                      LDA
C              THE VALUE OF NQ IS OBTAINED IN THE CALLING PROGRAM      LDA
C              BY  NQ = JSTART.                                        LDA
C                                                                      LDA
C     YL       - ARRAY OF NL = N - NY VARIABLES WHICH APPEAR LINEARLY. LDA
C                THE USER SUPPLIES INITIAL VALUES FOR THESE VARIABLES. LDA
C     T        - CURRENT VALUE OF THE INDEPENDENT VARIABLE (TIME)      LDA
C     TEND     - END TIME                                              LDA
C     N        - TOTAL NUMBER OF VARIABLES                             LDA
C     NY       - NUMBER OF DIFFERENTIAL EQUATIONS AND NONLINEAR        LDA
C                VARIABLES.                                            LDA
C     M        - NUMBER OF VARIABLES INCLUDED IN THE ERROR TEST.       LDA
C                THIS NUMBER CAN BE NO GREATER THAN NY.  IF IT IS      LDA
C                GREATER THAN NY, NY VARIABLES ARE USED IN THE ERROR   LDA
C                TEST.                                                 LDA
C     JSTART   - INPUT AND OUTPUT INDICATOR.                           LDA
C                ON INPUT JSTART HAS THE FOLLOWING MEANINGS.           LDA
C                     <0  THIS INDICATES A RE-START FROM A PREVIOUS    LDA
C                         POINT FOLLOWING TERMINATION OF THE RUN OR    LDA
C                         SOLUTION OF ANOTHER PROBLEM DURING THE SAME LDA
C                         RUN.  PARAMETERS IN THE CALLING SEQUENCE     LDA
C                         MUST HAVE BEEN PRESERVED FROM THE PREVIOUS   LDA
C                         USE, PARTICULARLY THE ARRAYS                 LDA
C                             SAVE, YLSV, ESV, AND PW.                 LDA
C                         THESE ARRAYS MUST BE SAVED AFTER A CALL      LDA
C                         TO SUBROUTINE LDASAV, WHICH ALSO SAVES       LDA
C                         NECESSARY PARAMETERS INTERNAL TO LDASUB.     LDA
C                     =0  INDICATES AN INITIAL CALL TO LDASUB.  THE    LDA
C                         ROUTINE INITIALIZES ITSELF, SCALES THE       LDA
C                         DERIVATIVES IN Y(2,I) AND THEN PERFORMS THE LDA
C                         INTEGRATION UNTIL T > TEND.                  LDA
C                     >0  INDICATES THE SOLUTION IS TO BE CONTINUED.   LDA
C                         AFTER THE INITIAL ENTRY IT IS NEITHER        LDA
C                         DESIRABLE NOR NECESSARY TO RE-ENTER WITH     LDA
C                         JSTART = 0, SINCE THIS RE-INITIALIZES        LDA
C                         THE CODE, BEGINNING WITH A FIRST ORDER       LDA
C                         METHOD AGAIN.                                LDA
C                ON OUTPUT, JSTART IS SET TO THE VALUE OF NQ, THE      LDA
C                ORDER OF THE FORMULA CURRENTLY BEING USED.            LDA
C     KFLAG    - THE COMPLETION CODE INDICATOR, WITH THE FOLLOWING     LDA
C                MEANINGS                                              LDA
C                     +1  THE INTEGRATION WAS SUCCESSFUL               LDA
C                     -1  ERROR TEST FAILURE FOR H > HMIN              LDA
C                     -3  CORRECTOR FAILED TO CONVERGE FOR H > HMIN    LDA
C                     -4  CORRECTOR FAILED TO CONVERGE FOR FIRST       LDA
C                         ORDER METHOD                                 LDA
C                     -5  ERROR RETURN FROM SUBROUTINE NLITSL          LDA
C     MAXOR    - MAXIMUM ORDER DERIVATIVE THAT SHOULD BE USED IN THE   LDA
```

```
C               METHOD.  IT MUST BE NO GREATER THAN SIX.  IF IT IS
C               GREATER THAN SIX, THE MAXIMUM ORDER USED WILL BE SIX.
C       IPRT    - INTERNAL PRINT CONTROL INDICATOR
C                     = 0  NO PRINT
C                     > 0  PRINT COUNTERS, STEPSIZE, CURRENT TIME
C                          AND VALUES OF DEPENDENT VARIABLES AT
C                          EACH STEP.
C       H       - CURRENT STEPSIZE.  AN INITIAL VALUE MUST BE SUPPLIED
C               BUT NEED NOT BE THE ONE WHICH WILL BE USED, SINCE THE
C               SUBROUTINE WILL CHOOSE A SMALLER ONE IF NECCESSARY TO
C               KEEP THE ERROR PER STEP SMALLER THAN THE SPECIFIED
C               VALUE.  IT IS BETTER TO UNDERESTIMATE THE INITIAL
C               STEPSIZE THAN TO OVERESTIMATE IT.  THE STEPSIZE IS
C               NORMALLY NOT CHANGED BY THE USER.
C       HMIN    - MINIMUM STEPSIZE ALLOWED
C       HMAX    - MAXIMUM STEPSIZE ALLOWED
C       RMSEPS  - THE ERROR TEST CONSTANT.  THE ROOT-MEAN-SQUARE OF
C               THE SINGLE STEP ERROR ESTIMATES,  ER(I), DIVIDED BY
C               YMAX(I) = (MAXIMUM TO CURRENT TIME OF Y(I)) MUST BE
C               LESS THAN RMSEPS.  THE STEPSIZE AND/OR ORDER ARE
C               VARIED TO ACHIEVE THIS.
C       SAVE    - AN ARRAY OF LENGTH AT LEAST 7*NY
C       YLSV    - AN ARRAY OF LENGTH AT LEAST NL
C       YMAX    - A VECTOR OF LENGTH NY WHICH CONTAINS THE MAXIMUM
C               OF EACH Y SEEN SO FAR.  ON THE FIRST CALL, THESE WILL
C               BE INITIALIZED AS YMAX(I) = MAX(1,|Y(1,I)|)
C       ER      - A VECTOR OF LENGTH NY
C       ESV     - A VECTOR OF LENGTH NY
C       F1      - A VECTOR OF LENGTH N = NY + NL
C       DY      - A VECTOR OF LENGTH N = NY + NL
C       PW      - AN ARRAY IN WHICH THE  J  MATRIX COMPUTED
C               IN SUBROUTINE JACMAT WILL BE STORED.  SIZE WHICH
C               MUST BE ALLOWED IS DETERMINED BY THE STORAGE TECH-
C               NIQUE USED FOR IT, BUT NORMALLY WON'T BE MORE THAN
C               N**2 + 2*N LOCATIONS, THE LATTER 2*N BEING REQUIRED
C               BY THE LINEAR EQUATION SOLVER.
C
C     ----------------------------------------------------------------
      INTEGER*2 IBP,ITYPE,JA,JB,NAME,NELCON
      DIMENSION BIGH12(1), BIGH3(1), BIGH5(1)
      DIMENSION BIGH4(1)
      DIMENSION BIGK(1)
      DIMENSION BIGF(1)
      DIMENSION IBP(1), ITYPE(1)
      DIMENSION JA(1), JB(1), NAME(1)
      DIMENSION NELCON(11,1)
      DIMENSION RR(1),Z(1)
      DIMENSION YIT(1)
```

```
      DIMENSION Y(7,1), YL(1), SAVE(7,1), YMAX(1), ER(1), YLSV(1), F1(1)
     1, PERT(6,3), COF(21), ESV(1), DY(1), PW(1), SAV(1), A(29)
      EQUIVALENCE (A(8),BND), (A(9),BR), (A(10),E), (A(11),EDWN),
     1(A(12),ENQ1), (A(13),ENQ2), (A(14),ENQ3), (A(15),EPS), (A(16),EUP)
     2,(A(17),HNEW), (A(18),PEPSH), (A(19),IDOUB), (A(20),IWEVAL),
     3(A(21),K), (A(22),LCOPYL), (A(23),LCOPYY), (A(24),MAXDER),
     4(A(25),M1), (A(26),NL), (A(27),NQ), (A(28),NS), (A(29),NW)
C
C     ------------------------------------------------------------------
C
C     THE COEFFICIENTS IN THE PERT ARRAY ARE USED FOR ERROR TESTING AND
C     CHANGING STEPSIZE AND NEED TO BE ACCURATE TO ONLY A FEW DIGITS.
C
C     ------------------------------------------------------------------
      DATA PERT/4.,9.,16.,25.,36.,49.,9.,16.,25.,36.,49.,64.,1.,1.,.25,
     12.7889E-2,1.70569E-3,6.83929E-5/
C     ------------------------------------------------------------------
C
C     THE ENTRIES IN THE COF ARRAY ARE THE COEFFICIENTS FOR THE STIFFLY
C     STABLE METHODS USED IN THIS PROGRAM AND ARE TO BE THE MACHINE
C     PRECISION EQUIVALENTS OF THE FOLLOWING CONSTANTS.
C
C     -1
C     -3/2 , -1/2
C     -11/6 , -1 , -1/6
C     -25/12 , -35/24 , -5/12 , -1/24
C     -137/60 , -15/8 , -17/24 , -1/8 , -1/120
C     -147/60, -203/90 , -49/48 , -35/144 , -7/240 , -1/720
C
C     ------------------------------------------------------------------
      DATA COF/-1.,-1.5,-.5,-1.833333,-1.,-.1666667,-2.083333,-1.458333,
     1-.4166667,-.04166667,-2.283333,-1.875,-.7083333,-.125,-.008333333,
     2-2.45,-2.255556,-1.020833,-.2430556,-.02916667,-.001388889/
      IF (JSTART) 100,110,150
C     ------------------------------------------------------------------
C     IF THIS IS A RESTART ENTRY, RESTORE Y AND YL FROM THE SAVE AND
C     YLSV ARRAYS, WHERE THEY WERE SAVED BY A PREVIOUS CALL TO LDASAV.
C     ------------------------------------------------------------------
  100 CALL COPYZ (Y,SAVE,LCOPYY)
      CALL COPYZ (YL,YLSV,LCOPYL)
      GO TO 150
C     ------------------------------------------------------------------
C     IF THIS IS THE FIRST CALL, INITIALIZE YMAX, SCALE DERIVATIVES, AND
C     INITIALIZE INDICATORS AND SET ORDER TO ONE.
C     FOR DOUBLE PRECISION, SET LCOPYL = 14*NY AND LCOPYL = 2*NL IF
C     SUBROUTINE COPYZ IS IN SINGLE PRECISION.
C     ------------------------------------------------------------------
  110 NL = N-NY
```

```
      LOCPYY = 7*NY                                                     LDA 1820
      LOCPYL = NL                                                       LDA 1830
      M1 = MINO(M,NY)                                                   LDA 1840
      EPS = SQRT(FLOAT(M1))*RMSEPS                                      LDA 1850
      MAXDER = MINO(MAXOR,6)                                            LDA 1860
      IF (IPRT.LE.0) GO TO 120                                          LDA 1870
      PRINT 3, N,NL,RMSEPS,TEND,H                                       LDA 1880
      PRINT 4                                                           LDA 1890
  120 NS = 0                                                            LDA 1900
      NW = 0                                                            LDA 1910
C                                                                       LDA 1920
      DO 130 J=1,NY                                                     LDA 1930
      YMAX(J) = AMAX1(1.,ABS(Y(1,J)))                                   LDA 1940
  130 Y(2,J) = Y(2,J)*H                                                 LDA 1950
C                                                                       LDA 1960
      NQ = 1                                                            LDA 1970
      BR = 1.                                                           LDA 1980
      ASSIGN 190 TO IRET                                                LDA 1990
C     -----------------------------------------------------------------LDA 2000
C     SET COEFFICIENTS FOR THE ORDER CURRENTLY BEING USED.              LDA 2010
C     E IS A TEST FOR ERRORS OF THE CURRENT ORDER NQ                    LDA 2020
C     EUP IS TO TEST FOR INCREASING THE ORDER, EDWN FOR DECREASING THE  LDA 2030
C     ORDER.                                                            LDA 2040
C     -----------------------------------------------------------------LDA 2050
  140 K = NQ*(NQ-1)/2                                                   LDA 2060
      CALL COPYZ (A(2),COF(K+1),NQ)                                     LDA 2070
      K = NQ+1                                                          LDA 2080
      IDOUB = NQ                                                        LDA 2090
      ENQ1 = .5/NQ                                                      LDA 2100
      ENQ2 = .5/K                                                       LDA 2110
      ENQ3 = .5/(NQ+2)                                                  LDA 2120
      PEPSH = EPS**2                                                    LDA 2130
      E = PERT(NQ,1)*PEPSH                                              LDA 2140
      EUP = PERT(NQ,2)*PEPSH                                            LDA 2150
      EDWN = PERT(NQ,3)*PEPSH                                           LDA 2160
      BND = (EPS*ENQ3)**2                                               LDA 2170
      IWEVAL = 1                                                        LDA 2180
      GO TO IRET, (190,200,490,570)                                     LDA 2190
  150 IF (H.EQ.HNEW) GO TO 190                                          LDA 2200
C     -----------------------------------------------------------------LDA 2210
C     IF CALLER HAS CHANGED H, RESCALE DERIVATIVES TO REFLECT THAT HNEW LDA 2220
C     WAS USED ON THE LAST CALL.                                        LDA 2230
C     -----------------------------------------------------------------LDA 2240
      R = H/HNEW                                                        LDA 2250
      ASSIGN 190 TO IRET                                                LDA 2260
      GO TO 610                                                         LDA 2270
C     -----------------------------------------------------------------LDA 2280
C     SET JSTART TO NQ, THE CURRENT ORDER OF THE METHOD, BEFORE EXIT,   LDA 2290
```

```
C      AND SAVE THE CURRENT STEPSIZE IN HNEW.                                LDA 2300
C      ---------------------------------------------------------------------- LDA 2310
  160  JSTART = NQ                                                          LDA 2320
       HNEW = H                                                             LDA 2330
       RETURN                                                               LDA 2340
  170  NS = NS+1                                                            LDA 2350
       IF (IPRT.LE.0) GO TO 180                                             LDA 2360
C      ----------------------------------------------------------------------LDA 2370
C      PRINT DATA IF DESIRED BY USER                                        LDA 2380
C      ----------------------------------------------------------------------LDA 2390
       PRINT 1, NS,NW,NQ,H,T,(Y(1,I),I=1,NY)                                LDA 2400
  180  CONTINUE                                                             LDA 2420
       IF (KFLAG.LT.0) GO TO 160                                            LDA 2430
       IF (T.GE.TEND) GO TO 160                                             LDA 2440
C      ----------------------------------------------------------------------LDA 2450
C      TAKE ANOTHER STEP IF T < TEND                                        LDA 2460
C      ----------------------------------------------------------------------LDA 2470
       JSTART = 1                                                           LDA 2480
C      ----------------------------------------------------------------------LDA 2490
C      SAVE DATA FOR TRIAL WITH A SMALLER TIMESTEP IF THIS STEP FAILS       LDA 2500
C      ----------------------------------------------------------------------LDA 2510
  190  CALL COPYZ (SAVE,Y,LCOPYY)                                           LDA 2520
       CALL COPYZ (YLSV,YL,LCOPYL)                                          LDA 2530
       RACUM = 1.                                                           LDA 2540
       KFLAG = 1                                                            LDA 2550
       HOLD = H                                                             LDA 2560
       NQOLD = NQ
       TOLD = T                                                             LDA 2580
  200  T = T+H                                                              LDA 2590
       HINV = 1./H                                                          LDA 2600
C      ----------------------------------------------------------------------LDA 2610
C      COMPUTE PREDICTED VALUES BY EFFECTIVELY MULTIPLYING DERIVATIVE       LDA 2620
C      VECTOR BY PASCAL TRIANGLE MATRIX                                     LDA 2630
C      ----------------------------------------------------------------------LDA 2640
C                                                                           LDA 2650
       DO 210 J=2,K                                                         LDA 2660
       J3 = K+J-1                                                           LDA 2670
C                                                                           LDA 2680
       DO 210 J1=J,K                                                        LDA 2690
       J2 = J3-J1                                                           LDA 2700
C                                                                           LDA 2710
       DO 210 I=1,NY                                                        LDA 2720
  210  Y(J2,I) = Y(J2,I)+Y(J2+1,I)                                          LDA 2730
C                                                                           LDA 2740
C                                                                           LDA 2750
       DO 220 I=1,NY                                                        LDA 2760
  220  ER(I) = 0.                                                           LDA 2770
C                                                                           LDA 2780
```

```
C     ----------------------------------------------------------------------LDA 2790
C     DO UP TO THREE CORRECTOR ITERATIONS.  CONVERGENCE IS OBTAINED WHENLDA 2800
C     CHANGES ARE LESS THAN BND WHICH IS DEPENDENT ON THE ERROR TEST    LDA 2810
C     CONSTANT.  THE SUM OF CORRECTIONS IS ACCUMULATED IN ER(I).  IT IS LDA 2820
C     EQUAL TO THE K-TH DERIVATIVE OF Y TIMES H**K/(K-FACTORIAL*A(K)),  LDA 2830
C     AND THUS IS PROPORTIONAL TO THE ACTUAL ERRORS TO THE LOWEST POWER LDA 2840
C     OF H PRESENT, WHICH IS H**K.                                      LDA 2850
C     ----------------------------------------------------------------------LDA 2860
C                                                                       LDA 2870
      DO 270 L=1,3                                                      LDA 2880
      CALL DIFFUN (Y,YL,T,HINV,DY,BIGH12,BIGH3,BIGH4,BIGH5,BIGK,BIGF,
     1 IEP,ITYPE,JA,JB,NAME,NELCON,RR,Z,YIT)
      IF (IWEVAL.LT.1) GO TO 230                                        LDA 2900
C     ----------------------------------------------------------------------LDA 2910
C     IF THERE HAS BEEN A CHANGE OF ORDER OR THERE HAS BEEN TROUBLE     LDA 2920
C     WITH CONVERGENCE, PW IS RE-EVALUATED PRIOR TO STARTING THE        LDA 2930
C     CORRECTOR ITERATION.  IWEVAL IS THEN SET TO -1 AS AN INDICATOR    LDA 2940
C     THAT IT HAS BEEN DONE.  NEWPW IS SET NONZERO TO INDICATE TO       LDA 2950
C     SUBROUTINE NUITSL THAT A NEW PW HAS BEEN PROVIDED.                LDA 2960
C     ----------------------------------------------------------------------LDA 2970
      CALL JACMAT (Y,YL,T,HINV,A(2),N,NY,EPS,DY,F1,PW,BIGH12,BIGH3,
     1 BIGH4,BIGH5,BIGK,IBF,ITYPE,JA,JB,NAME,NELCON,RR,Z)
      KFLAG = 1                                                         LDA 2990
      IWEVAL = -1                                                       LDA 3000
      NW = NW+1                                                         LDA 3010
      NEWPW = 1                                                         LDA 3020
  230 CALL NUITSL (PW,DY,F1,N,NY,EPS,YMAX,NEWPW,KRRET,BIGH3,ITYPE,
     1 JA,JB,NAME,NELCON)
      IF (KRRET.NE.0) GO TO 600                                         LDA 3040
      IF (NL.LE.0) GO TO 250                                            LDA 3050
C                                                                       LDA 3060
      DO 240 I=1,NL                                                     LDA 3070
  240 YL(I) = YL(I)-F1(I+NY)                                            LDA 3080
C                                                                       LDA 3090
  250 CONTINUE                                                          LDA 3100
      DEL = 0.                                                          LDA 3110
C                                                                       LDA 3120
      DO 260 I=1,NY                                                     LDA 3130
      Y(1,I) = Y(1,I)-F1(I)                                             LDA 3140
      Y(2,I) = Y(2,I)+A(2)*F1(I)                                        LDA 3150
      ER(I) = ER(I)+F1(I)                                               LDA 3160
      DEL = DEL+(F1(I)/AMAX1(YMAX(I),ABS(Y(1,I))))**2                   LDA 3170
  260 CONTINUE                                                          LDA 3180
C                                                                       LDA 3190
      IF (L.GE.2) BR = AMAX1(.9*BR,DEL/DEL1)                            LDA 3200
      DEL1 = DEL                                                        LDA 3210
      IF (AMIN1(DEL,BR*DEL*2.).LE.BND) GO TO 330                        LDA 3220
  270 CONTINUE                                                          LDA 3230
```

```
C
C     -----------------------------------------------------------------
C     THE CORRECTION ITERATION FAILED TO CONVERGE IN 3 TRIES.  VARIOUS
C     POSSIBILITIES ARE CHECKED FOR.  IF H IS ALREADY HMIN AND PW HAS
C     ALREADY BEEN RE-EVALUATED, A NO CONVERGENCE EXIT IS TAKEN.
C     OTHERWISE THE MATRIX PW IS RE-EVALUATED AND/OR (IN THAT ORDER) THE
C     STEP IS REDUCED TO TRY AND GET CONVERGENCE.
C     -----------------------------------------------------------------
      T = TOLD
      IF (IWEVAL) 280,300,290
  280 IF (H.LE.HMIN*1.00001) GO TO 310
  290 RACUM = RACUM*.25
  300 CONTINUE
      GO TO 560
  310 KFLAG = -3
C     -----------------------------------------------------------------
C     RESTORE Y AND YL AFTER CONVERGENCE FAILURE
C     -----------------------------------------------------------------
  320 CALL COPYZ (Y,SAVE,LCOPYY)
      CALL COPYZ (YL,YLSV,LCOPYL)
      H = HOLD
      NQ = NQOLD
      GO TO 170
C     -----------------------------------------------------------------
C     THE CORRECTOR CONVERGED, SO NOW THE ERROR TEST IS MADE.
C     -----------------------------------------------------------------
  330 D = 0.
C
      DO 340 I=1,M1
      YM = AMAX1(ABS(Y(1,I)),YMAX(I))
  340 D = D+(ER(I)/YM)**2
C
      IWEVAL = 0
      IF (D.GT.E) GO TO 380
C     -----------------------------------------------------------------
C     THE ERROR TEST IS OKAY, SO THE STEP IS ACCEPTED.  IF IDOUB
C     NOW BECOMES NEGATIVE, A TEST IS MADE TO SEE IF THE STEP SIZE
C     CAN BE INCREASED AT THIS ORDER OR ONE HIGHER OR ONE LOWER.
C     THE CHANGE IS MADE ONLY IF THE STEP CAN BE INCREASED BY AT
C     LEAST 10%.  IDOUB IS SET TO NQ TO PREVENT FURTHER TESTING
C     FOR A WHILE.  IF NO CHANGE IS MADE, IDOUB IS SET TO 5.
C     -----------------------------------------------------------------
      IF (K.LT.3) GO TO 360
C
      DO 350 J=3,K
C
      DO 350 I=1,NY
  350 Y(J,I) = Y(J,I)+A(J)*ER(I)
```

```
C
  360 KFLAG = 1
      IDOUB = IDOUB-1
      IF (IDOUB) 410,370,510
  370 CALL COPYZ (ESV,ER,M1)
      GO TO 510
C     ----------------------------------------------------------------------
C     THE ERROR TEST FAILED.  IF JSTART = 0, THE DERIVATIVES IN THE
C     SAVE ARRAY ARE UPDATED.  TESTS ARE THEN MADE TO FIX THE STEPSIZE
C     AND PERHAPS REDUCE THE ORDER.  AFTER RESTORING AND SCALING THE
C     Y VARIABLES, THE STEP IS RETRIED.
C     ----------------------------------------------------------------------
  380 IF (JSTART.GT.0) GO TO 400
C
      DO 390 I=1,NY
  390 SAVE(2,I) = Y(2,I)
C
  400 KFLAG = KFLAG-2
      IF (H.LE.HMIN) GO TO 550
      T = TOLD
      IF (KFLAG.LE.-5) GO TO 530
  410 PR2 = (D/E)**ENQ2*1.2
      L = 0
      IF (NQ.LE.1) GO TO 430
      D = 0.
C
      DO 420 J=1,M1
      YM = AMAX1(ABS(Y(1,J)),YMAX(J))
  420 D = D+(Y(K,J)/YM)**2
C
      PR1 = (D/EDWN)**ENQ1*1.3
      IF (PR1.GE.PR2) GO TO 430
      PR2 = PR1
      L = -1
  430 IF (KFLAG.LT.0.OR.NQ.GE.MAXDER) GO TO 450
      D = 0
C
      DO 440 J=1,M1
      YM = AMAX1(ABS(Y(1,J)),YMAX(J))
  440 D = D+((ER(J)-ESV(J))/YM)**2
C
      PR1 = (D/EUP)**ENQ3*1.4
      IF (PR1.GE.PR2) GO TO 450
      PR2 = PR1
      L = 1
  450 R = 1./AMAX1(PR2,1.E-5)
      IF (KFLAG.LT.0.OR.R.GE.1.1) GO TO 460
      IDOUB = 9
```

```
      GO TO 510
  460 NEWQ = NQ+L
      K = NEWQ+1
      IF (NEWQ.LE.NQ) GO TO 480
      R1 = A(NEWQ)/FLOAT(NEWQ)
C
      DO 470 J=1,NY
  470 Y(K,J) = ER(J)*R1
C
  480 CONTINUE
C     ---------------------------------------------------------------------
C     IF THE STEP WAS OKAY, SCALE THE Y VARIABLES IN ACCORDANCE
C     WITH THE NEW VALUE OF H.  IF KFLAG < 0, HOWEVER, USE THE
C     SAVED VALUES (IN SAVE AND YLSV).  IN EITHER CASE, IF THE ORDER
C     HAS CHANGED IT IS NECESSARY TO FIX CERTAIN PARAMETERS BY CALLING
C     THE PROGRAM SEGMENT AT STATEMENT NUMBER 140.
C     ---------------------------------------------------------------------
      IDOUB = NQ
      IF (NEWQ.EQ.NQ) GO TO 490
      NQ = NEWQ
      ASSIGN 490 TO IRET
      GO TO 140
  490 IF (KFLAG.GT.0) GO TO 500
      RACUM = RACUM*R
      GO TO 560
  500 R = AMAX1(AMIN1(HMAX/H,R),HMIN/H)
      H = H*R
      IWEVAL = 1
      ASSIGN 510 TO IRET
      GO TO 610
C
  510 DO 520 I=1,M1
  520 YMAX(I) = AMAX1(ABS(Y(1,I)),YMAX(I))
C
      GO TO 170
C     ---------------------------------------------------------------------
C     THE ERROR TEST HAS NOW FAILED THREE TIMES, SO THE DERIVATIVES ARE
C     IN BAD SHAPE.  RETURN TO FIRST ORDER METHOD AND TRY AGAIN.  OF
C     COURSE, IF NQ = 1 ALREADY, THEN THERE IS NO HOPE AND WE EXIT WITH
C     KFLAG = -4.
C     ---------------------------------------------------------------------
  530 IF (NQ.EQ.1) GO TO 540
      NQ = 1
      IDOUB = 1
      ASSIGN 570 TO IRET
      GO TO 140
  540 NQOLD = 1
      KFLAG = -4
```

```
      GO TO 320
  550 KFLAG = -1
      GO TO 170
C     ------------------------------------------------------------------
C     THIS SECTION RESTORES THE SAVED VALUES OF Y AND YL, SCALING THE
C     Y DERIVATIVES AS NECESSARY, AND THEN RETURNS TO THE PREDICTOR LOOP
C     ------------------------------------------------------------------
  560 H = HOLD*RACUM
      H = AMAX1(HMIN,AMIN1(H,HMAX))
  570 RACUM = H/HOLD
      R1 = 1.
      DO 580 J=2,K
      R1 = R1*RACUM
C
      DO 580 I=1,NY
  580 Y(J,I) =  SAVE(J,I)*R1
C
C
      DO 590 I=1,NY
  590 Y(1,I) = SAVE(1,I)
C
      CALL COPYZ (YL,YLSV,LOCPYL)
      IWEVAL = 1
      GO TO 200
  600 KFLAG = -5
      GO TO 160
C     ------------------------------------------------------------------
C     THIS SECTION SCALES THE Y DERIVATIVES BY R**J
C     -----------------------------------------------------------------
  610 R1 = 1.
C
      DO 620 J=2,K
      R1 = R1*R
C
      DO 620 I=1,NY
  620 Y(J,I) = Y(J,I)*R1
C
      GO TO IRET, (190,510)
C     ------------------------------------------------------------------
C     THIS SECTION ALLOWS FOR RESTARTS AFTER SOLVING ANOTHER PROBLEM, OR
C     HAVING TERMINATED THE CURRENT COMPUTER RUN.  SUBROUTINE LDASAV
C     SAVES THE NECESSARY VALUES WHICH ARE INTERNAL TO LDASUB.  FOR
C     DOUBLE PRECISION, WITH COPYZ IN SINGLE PRECISION, THE NUMBER OF
C     LOCATIONS TO BE SAVED AND RESTORED, LCOPYS AND LOCPYR, MUST BE
C     SET TO 58.
C     IT IS ASSUMED THAT IN ADDITION TO THE VARIABLES IN THE ARRAY A
C     SAVED BY CALLING LDASAV, THE USER ALSO SAVES THE ARRAYS SAVE,
C     YLSV, YMAX, ESV, AND PW.
```

```
C
C     TO RESTART THE USER FIRST CALLS LDARST TO RESTORE THE VALUES SAVED
C     BY LDASAV, THEN RE-ENTERS LDASUB WITH JSTART < 0, AND WITH THE
C     OTHER PARAMETERS THE SAME AS RETURNED FROM THE LAST ENTRY TO
C     LDASUB, PARTICULARLY THOSE ARRAYS MENTIONED ABOVE.
C     ----------------------------------------------------------------
      ENTRY LDASAV(SAV)
      LCOPYS = 29
      CALL COPYZ (SAV,A,LCOPYS)
      CALL COPYZ (SAVE,Y,LCOPYY)
      CALL COPYZ (YLSV,YL,LCOPYL)
      RETURN
C
      ENTRY LDARST(SAV)
      LCOPYR = 29
      CALL COPYZ (A,SAV,LCOPYR)
      RETURN
C
C     ----------------------------------------------------------------
C
C
    1 FORMAT (2I5,I2,1P2E10.2,7E14.6/(32X,7E14.6))
    2 FORMAT (32X,1P7E14.6)
    3 FORMAT ('1  N =',I3,'   NL =',I3,'   RMSEPS =',1PE9.2,'   TENC ='
     1  ,E9.2,'   H =',E9.2//)
    4 FORMAT ('    NS   NW Q    H',8X,'T    ',8X,'Y(1,*) AND YL(*)'//)
      END
```

```
      SUBROUTINE SDESOL (Y,YL,T,TEND,NY,NL,M,JSKF,MAXDER,IPRT,H,HMIN,    SDE
     1 HMAX,RMSEPS,W,BIGH12,BIGH3,BIGH5,BIGK,BIGF,IBP,ITYPE,JA,JB,NAME,
     2 NELCON,R,Z,YIT,BIGH4)
C                                                                        SDE
C     -------------------------------------------------------------------SDE
C                                                                        SDE
C     SUBROUTINE SDESOL IS A DRIVER ROUTINE FOR SUBROUTINE LDASUB.       SDE
C     ITS PURPOSE IS TO SET UP THE NECESSARY REFERENCES TO A LARGE       SDE
C     BLOCK OF AUXILLARY STORAGE, AND  OBTAIN INITIAL VALUES OF          SDE
C     DERIVATIVES.                                                       SDE
C     THE CALLING SEQUENCE FOR SDESOL IS                                 SDE
C                                                                        SDE
C CALL SDESOL(Y,YL,T,TEND,NY,NL,M,JSKF,MAXDER,IPRT,H,HMIN,HMAX,RMSEPS,W)SDE
C                                                                        SDE
C     WHERE THE PARAMETERS ARE DEFINED AS FOLLOWS.                       SDE
C                                                                        SDE
C        Y         - ARRAY DIMENSIONED (7,NY).  THIS ARRAY CONTAINS THE  SDE
C                    DEPENDENT VARIABLES AND THEIR SCALED DERIVATIVES.   SDE
C                    Y(J+1,I) CONTAINS THE J-TH DERIVATIVE OF THE I-TH VARSDE
C                    IABLE TIMES H**J/J-FACTORIAL, WHERE H IS THE CURRENT SDE
C                    STEPSIZE.  ON FIRST ENTRY THE CALLER SUPPLIES THE   SDE
C                    INITIAL VALUES OF EACH VARIABLE IN Y(1,I).  ON SUB- SDE
C                    SEQUENT ENTRIES IT IS ASSUMED THE ARRAY HAS NOT     SDE
C                    BEEN CHANGED.  TO INTERPOLATE TO NON-MESH POINTS,   SDE
C                    THESE VALUES CAN BE USED AS FOLLOWS.  IF H IS THE   SDE
C                    CURRENT STEPSIZE AND VALUES AT TIME  T+E  ARE       SDE
C                    NEEDED, LET  S = E/H  AND THEN                      SDE
C                                                                        SDE
C                                                    JS                  SDE
C                            I-TH VARIABLE AT  T+E IS  SUM Y(J+1,I)*S**J SDE
C                                                    J=0                 SDE
C                                                                        SDE
C                    THE VALUE OF JS IS OBTAINED IN THE CALLING PROGRAM  SDE
C                    BY  JS = IABS(JSKF/10)                              SDE
C        YL        - ARRAY OF NL VARIABLES WHICH APPEAR LINEARLY.        SDE
C        T         - CURRENT VALUE OF THE INDEPENDENT VARIABLE (TIME)    SDE
C        TEND      - END TIME                                            SDE
C        NY        - NUMBER OF DIFFERENTIAL EQUATIONS AND NONLINEAR      SDE
C                    VARIABLES.                                          SDE
C        NL        - NUMBER OF LINEAR VARIABLES                          SDE
C        M         - NUMBER OF VARIABLES INCLUDED IN THE ERROR TEST      SDE
C        JSKF      - AN INDICATOR USED BOTH ON INPUT AND OUTPUT          SDE
C                    ON INPUT, JSKF = -1 INDICATES A RESTART CALL TO     SDE
C                    SDESOL.  JSKF = 0 INDICATES AN INITIAL CALL TO      SDE
C                    SDESOL.  JSKF > 0 INDICATES A CONTINUATION OF THE   SDE
C                    PREVIOUS CALL TO SDESOL.  JSKF < -1 MAY HAVE RESULTEDSDE
C                    FROM THE USER NEGLECTING TO TEST FOR ERROR RETURNS  SDE
C                    FROM SDESOL.  BECAUSE OF THIS POSSIBILITY, JSKF < -1 SDE
```

```
C                    RESULTS IN TERMINATION OF THE RUN WITH THE
C                    APPROPRIATE COMMENT.
C                    ON OUTPUT, JSKF CONSISTS OF TWO DIGITS AND SIGN,
C                    + OR - QP.  Q IS THE ORDER OF THE FORMULA CURRENTLY
C                    BEING USED.  P INDICATES THE TYPE OF RETURN, AS
C                    FOLLOWS.
C                    JSKF > 0, P = 1 IS THE NORMAL RETURN
C                    JSKF < 0 IS AN ERROR RETURN, WITH THE FOLLOWING
C                    MEANINGS.
C                         P = 1  ERROR TEST FAILURE FOR H > HMIN
C                         P = 3  CORRECTOR FAILED TO CONVERGE FOR H > HMIN
C                         P = 4  CORRECTOR FAILED TO CONVERGE FOR FIRST
C                                ORDER METHOD
C                         P = 5  ERROR RETURN FROM SUBROUTINE NLITSL
C                         P = 6  ERROR RETURN FROM SUBROUTINE DERVAL
C           MAXDER  - MAXIMUM ORDER DERIVATIVE THAT SHOULD BE USED IN
C                     METHOD.  IT MUST BE NO GREATER THAN SIX.
C           IPRT    - INTERNAL PRINT CONTROL INDICATOR FOR LEASUB.
C                         IPRT = 0  NO PRINT
C                         IPRT > 0  PRINT COUNTERS, STEPSIZE, CURRENT TIME
C                                   AND VALUES OF DEPENDENT VARIABLES AT
C                                   EACH STEP.
C           H       - CURRENT STEPSIZE.  AN INITIAL VALUE MUST BE SUPPLIED
C                     BUT NEED NOT BE THE ONE WHICH MUST BE USED, SINCE THE
C                     SUBROUTINE WILL CHOSE A SMALLER ONE IF NECCESSARY TO
C                     KEEP THE ERROR PER STEP SMALLER THAN THE SPECIFIED
C                     VALUE.  IT IS BETTER TO UNDERESTIMATE THE INITIAL
C                     STEPSIZE THAN TO OVERESTIMATE IT.  THE STEPSIZE IS
C                     NORMALLY NOT CHANGED BY THE USER.
C           HMIN    - MINIMUM STEPSIZE ALLOWED
C           HMAX    - MAXIMUM STEPSIZE ALLOWED
C           RMSEPS  - THE ERROR TEST CONSTANT.  THE ROOT-MEAN-SQUARE OF
C                     THE SINGLE STEP ERROR ESTIMATES, ER(I), DIVIDED BY
C                     YMAX(I) = (MAXIMUM TO CURRENT TIME OF Y(I) ) MUST BE
C                     LESS THAN EPS.  THE STEPSIZE AND/OR THE ORDER
C                     ARE VARIED TO ACHIEVE THIS.
C           W       - SCRATCH STORAGE ARRAY.  MUST BE AT LEAST 13*NY + 5*NL
C                     LOCATIONS, PLUS THOSE REQUIRED FOR STORAGE OF THE
C                     MATRIX  PW (SEE DESCRIPTION OF SUBROUTINE JACMAT).
C                     THE STORAGE OF PW WILL NORMALLY REQUIRE NO MORE THAN
C                     N**2 + 2*N LOCATIONS, AND IF COMPACT STORAGE TECH-
C                     NIQUES ARE USED, CAN BE MUCH FEWER.
C
C     ----------------------------------------------------------------------
      INTEGER*2 IBP,ITYPE,JA,JB,NAME,NELCON
      DIMENSION Y(7,1), YL(1), W(1)
      DIMENSION BIGH12(1), BIGH3(1), BIGH5(1)
      DIMENSION BIGH4(1)
```

```
      DIMENSION BIGK(1)
      DIMENSION BIGF(1)
      DIMENSION IBP(1), ITYPE(1)
      DIMENSION JA(1), JB(1), NAME(1)
      DIMENSION NELCON(11,1)
      DIMENSION R(1), Z(1)
      DIMENSION YIT(1)
      IF (JSKF.GT.0) GO TO 120
      IF (JSKF.LT.-1) GO TO 140
      N = NY+NL
      IF (JSKF.LT.0) GO TO 110
C
C     IF THIS IS THE FIRST ENTRY, OBTAIN VALUES OF THE DERIVATIVES.
      CALL DERVAL (Y,YL,T,N,NY,W,KRETR,BIGH12,BIGH3,BIGH4,BIGH5,BIGK,
     1 BIGF,IBP,ITYPE,JA,JB,NAME,NELCON,R,Z,YIT)
      IF (KRETR.NE.0) GO TO 130
C
C     NOW SET UP STORAGE BLOCKS IN THE W ARRAY.  THIS NEEDS TO BE DONE
C     ONLY INITIALLY AND ON RESTARTS.
C
C        THE ARRAY   SAVE    STARTS AT LOCATION  1      IN THE W ARRAY
C        THE ARRAY   YLSV    STARTS AT LOCATION  NSVL   IN THE W ARRAY
C        THE ARRAY   YMAX    STARTS AT LOCATION  NYMAX  IN THE W ARRAY
C        THE ARRAY   ER      STARTS AT LOCATION  NER    IN THE W ARRAY
C        THE ARRAY   ESV     STARTS AT LOCATION  NESV   IN THE W ARRAY
C        THE ARRAY   F1      STARTS AT LOCATION  NF1    IN THE W ARRAY
C        THE ARRAY   DY      STARTS AT LOCATION  NDY    IN THE W ARRAY
C        THE MATRIX  PW      STARTS AT LOCATION  NPW    IN THE W ARRAY
C
  110 NSVL = 7*NY+1
      NYMAX = NSVL+NL
      NER = NYMAX+NY
      NESV = NER+NY
      NF1 = NESV+NY
      NDY = NF1+N
      NPW = NDY+N
  120 JS = JSKF
      CALL LDASUB (Y,YL,T,TEND,N,NY,M,JS,KF,MAXDER,IPRT,H,HMIN,HMAX,
     1RMSEPS,W,W(NSVL),W(NYMAX),W(NER),W(NESV),W(NF1),W(NDY),W(NPW),
     2BIGH12,BIGH3,BIGH4,BIGH5,BIGK,BIGF,IBP,ITYPE,JA,JB,NAME,NELCON,
     3R,Z,YIT)
C
C     CODE JSKF ON RETURN FROM LDASUB
C
      JSKF = ISIGN(JS*10+IABS(KF),KF)
      RETURN
  130 JSKF = -6
      RETURN
```

```
  14C PRINT 1, JSKF
      STOP                                                                      SDE 1310
C                                                                               SDE 1320
C                                                                               SDE 1330
    1 FCRMAT ('0IT IS AN ERROR TO ENTER SDESOL WITH JSKF = ',I10//              SDE 1340
     1 '   RUN HAS BEEN TERMINATED.')                                           SDE 1350
      END                                                                       SDE 1360
                                                                                SDE 1370
```

```
      SUBROUTINE COPYZ(S,Y,L)
      DIMENSION S(1),Y(1)
C     ------------------------------------------------------------------
C
C     THIS SUBROUTINE COPIES THE ARRAY Y, OF LENGTH L, INTO THE ARRAY S
C
C     ------------------------------------------------------------------
      IF(L.LE.0)RETURN
      DO 100 J=1,L
  100 S(J) = Y(J)
      RETURN
      END
```

```
      SUBROUTINE DERVAL (Y,YL,T,N,NY,W,KERET,BIGH12,BIGH3,BIGH4,BIGH5,
     1 BIGK,BIGF,IBP,ITYPE,JA,JB,NAME,NELCON,R,Z,YIT)
      INTEGER*2 IBP,ITYPE,JA,JB,NAME,NELCON
      DIMENSION BIGH12(1), BIGH3(1), BIGH5(1)
      DIMENSION BIGH4(1)
      DIMENSION BIGK(1)
      DIMENSION BIGF(1)
      DIMENSION IBP(1), ITYPE(1)
      DIMENSION JA(1), JB(1), NAME(1)
      DIMENSION NELCON(11,1)
      DIMENSION R(1),Z(1)
      DIMENSION YIT(1)
C
C     THIS SUBROUTINE CALCULATES THE INTIAL VALUES OF THE DERIVATIVES
C     IN THE GENERAL CASE.  IT IS WRITTEN SO THAT IT SHOULD WORK IF THE
C     FIRST NY EQUATIONS ALL INVOLVE DERIVATIVES.  IT ATTEMPTS TO SOLVE
C     THE FIRST NY EQUATIONS USING NEWTON'S METHOD, BUT SINCE IT TRIES
C     TO EVALUATE DF/DY' BY CALLING JACMAT IN SUCH A WAY AS TO MAKE THE
C     DF/DY TERM INSIGNIFICANT, IT IS POSSIBLE THAT IT MAY FAIL FOR THAT
C     REASON.  IT MAY FAIL FOR OTHER REASONS, AS WELL.  IF IT DOES FAIL
C     THE USER CAN SUPPLY HIS OWN VERSION OF DERVAL, OR MODIFY THIS
C     ROUTINE IN SUITABLE FASHION.  THIS ROUTINE ASSUMES THAT VALUES OF
C     THE LINEAR VARIABLES HAVE BEEN SUPPLIED PREVIOUSLY.  IF THOSE
C     MUST BE SOLVED FOR SIMULTANEOUSLY WITH THE DERIVATIVES, THE USER
C     MUST SUPPLY HIS OWN VERSION OF DERVAL.
C
C     THE CALLING SEQUENCE FOR THIS SUBROUTINE IS
C
C     CALL DERVAL(Y,YL,T,N,NY,W,KERET)
C
C     WHERE THE PARAMETERS ARE DEFINED AS FOLLOWS
C
C         Y        - SAME AS IN LDASUB AND SDESOL.  Y(1,I) CONTAINS THE
C                    INITIAL VALUES OF THE DEPENDENT VARIABLES.   THE
C                    VALUES OF THE DERIVATIVES ARE RETURNED IN Y(2,I).
C         YL       - SAME AS IN LDASUB AND SDESOL.  THE INITIAL VALUES OF
C                    THE LINEAR VARIABLES MUST BE SUPPLIED TO THIS VERSION
C         T        - INITIAL TIME
C         N        - SAME AS IN LDASUB, TOTAL NUMBER OF VARIABLES
C         NY       - SAME AS IN LDASUB, NUMBER OF DIFFERENTIAL EQUATIONS
C                    AND NONLINEAR VARIABLES
C         W        - SCRATCH ARRAY W FROM THE CALLING SEQUENCE OF SDESOL.
C                    THIS CAN BE USED AS NEEDED IN THIS SUBROUTINE.
C         KERET    - RETURN INDCATOR
C                          =0   NORMAL RETURN
C                          =1   ERROR RETURN
C
C     -----------------------------------------------------------------
```

```
      DIMENSION Y(7,1), YL(1), W(1)                                      DER  380
C                                                                        DER  390
      DO 100 I=1,NY                                                      DER  400
      W(2*N+I) = AMAX1(ABS(Y(1,I)),1.)                                   DER  410
  100 Y(3,I) = 0.                                                        DER  420
C                                                                        DER  430
      HINV = 16.**20                                                     DER  440
      KERET = 0                                                          DER  450
      EPS2 = NY/1.E5
      EPS = SQRT(EPS2)                                                   DER  470
C                                                                        DER  480
      DO 140 IT=1,10                                                     DER  490
C                                                                        DER  500
      DO 110 I=1,NY                                                      DER  510
  110 Y(2,I) = Y(3,I)/HINV                                               DER  520
C                                                                        DER  530
      CALL DIFFUN (Y,YL,T,HINV, W,BIGH12,BIGH3,BIGH4,BIGH5,BIGK,BIGF,
     1 IBP,ITYPE,JA,JB,NAME,NELCON,R,Z,YIT)
      CALL JACMAT (Y,YL,T,HINV,-1.,NY,NY,EPS,W,W(N+1),W(3*N+1),BIGH12,
     1 BIGH3,BIGH4,BIGH5,BIGK,IBP,ITYPE,JA,JB,NAME,NELCON,R,Z)
      NEWPW = 1                                                          DER  560
C                                                                        DER  570
      DO 120 I=1,NY                                                      DER  580
  120 W(I) = W(I)*HINV                                                   DER  590
C                                                                        DER  600
      CALL NUITSL (W(3*N+1),W,W(N+1),NY,NY,EPS,W(2*N+1),NEWPW,KRET,
     1 BIGH3,ITYPE,JA,JB,NAME,NELCON)
      IF (KRET.NE.0) GO TO 170                                           DER  620
      ER = 0.                                                            DER  630
C                                                                        DER  640
      DO 130 I=1,NY                                                      DER  650
      Y(3,I) = Y(3,I)-W(N+I)                                             DER  660
      W(2*N+I) =AMAX1(ABS(Y(3,I)),1.)
  130 ER = ER+(W(N+I)/AMAX1(ABS(Y(3,I)),1.))**2                          DER  670
C                                                                        DER  680
      IF (ER.LT.EPS2) GO TO 150                                          DER  690
  140 CONTINUE                                                           DER  700
      GO TO 170                                                          DER  720
C                                                                        DER  710
C                                                                        DER  730
  150 DO 160 I=1,NY                                                      DER  740
  160 Y(2,I) = Y(3,I)                                                    DER  750
C                                                                        DER  760
      RETURN                                                             DER  770
  170 KERET = 1                                                          DER  780
      RETURN                                                             DER  790
      END
```

LIST OF REFERENCES

1. Apostolakis, G. E., "Analytical Estimate of the Error in Conventional Point-Kinetic Reactivity Due to Spatial Effects," Nucl. Sci. Engr., 53, p. 141-152, 1974.

2. Yasinsky, J. B., "On the Use of Point Kinetics for the Analysis of Rod Ejection Accidents," Nucl. Sci. Engr., 39, p. 241-256, 1970.

3. Jackson, J. F. and Kastenburg, W. S., "Space-time Dynamics Studies in Large LMFBR's with Feedback," Nucl. Sci. Engr., 42, p. 278-294, 1970.

4. Nguyen, D. H. and D. Salinas, "Finite Element Solution of Space-Time Nonlinear Reactor Dynamics," Nucl. Sci. Engr., 60, p. 120-130, 1976.

5. Hanford Engineering Development Laboratory Report HEDL-TME 92-128, User's Guide for GAPCON: A Computer Program to Predict Fuel-to-Cladding Heat Transfer Coefficients in Oxide Fuel Pins, by G. R. Horn and F. E. Panisko, 1972.

6. Hanford Engineering Development Laboratory Report HEDL-TME 74-47, Melt III-A Neutronics Thermal-Hydraulic Computer Program for Fast Reactor Safety, v. 1, by A. S. Walter and others, 1974.

7. Hanford Engineering Development Laboratory Report HEDL-TME 75-50, An Analysis of the Unprotected Transient Overpower Accident in the FTR, by A. S. Walter and others, 1975.

8. Naval Postgraduate School Report NPS-69Zc-76111, An Optimal Compact Storage Scheme for Nonlinear Reactor Problems by FEM, by D. Salinas, D. H. Nguyen, and R. Franke, 1976.

9. Naval Postgraduate School Report NPS-53Fe-76051, A Program for the Numerical Solution of Large Sparse Systems of Algebraic and Implicitly Defined Stiff Differential Equations, by R. Franke, 1976.

10. Lamarsh, J. R., Introduction to Nuclear Reactor Theory, Addison, Wesley, 1972.

11. Salinas, D., D. H. Nugyen and T. W. Southworth, "Finite Element Solution of a Nonlinear Nuclear Reactor Dynamics Problem," International Conference on Computational Methods in Nonlinear Mechanics, Austin, Texas, p. 541-550, 1974.

12. Semenze, L. A., Lewis, E. E., and Rossow, E. C.,"The Application of the Finite Element Method to the Multi-group Neutron Diffusion Equation," Nucl. Sci. Engr., 47, p. 302-310, 1972.

13. Zienkiewicz, O. C., The Finite Element Method in Engineering Science, McGraw-Hill, 1970.

14. Zlamal, M.,"On the Finite Element Method," Numer. Meth., 12, p. 394-409, 1968.

15. Personal conversation with Dr. G. Cantin of the Naval Postgraduate School, 1976.

16. Cook, R. D., Concepts and Applications of Finite Element Analysis, Wiley, 1974.

17. Zienkiewicz, O. C. and C. J. Parekh, "Transient Field Problems: Two-Dimensional and Three-Dimensional Analysis by Isoparametric Finite Elements," Int. J. for Numer. Meth. in Engr., v. 2, p. 61-71, 1970.

18. Barsoum, R. S., "On the Use of Isoparametric Finite Elements in Linear Fracture Mechanics," Int. J. for Numer. Meth. in Engr., v. 10, no. 1, p. 25-37, 1976.

19. Belle, J., Uranium Dioxide: Properties and Nuclear Applications, Naval Reactors, Div. of Reactor Development, U. S. Atomic Energy Commission, p. 189, 1961.

20. Lew, G. T., A Three Dimensional Solution of the Transient Field Problem Using Isoparametric Finite Elements, Masters Thesis, Naval Postgraduate School, 1972.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 69 Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
4.	Dr. D. H. Nguyen (thesis advisor) 4625 Larchmont NE Albuquerque, New Mexico 87115	1
5.	Assoc. Professor D. Salinas (thesis advisor) Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
6.	Assoc. Professor R. Franke (second reader) Department of Mathematics Naval Postgraduate School Monterey, California 93940	1
7.	Professor Gilles Cantin Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
8.	LT R. E. Kasdorf, USN (student) 129 Redondo Ct. Marina, California 93933	2

168562

Thesis
K14495 Kasdorf
c.1
Finite element solution of t'e nonlinear coupled neutronic-energy equations for a fast reactor fuel cell.